UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

DISEÑO Y DESARROLLO DE UNA HERRAMIENTA PARA LA EXTRACCIÓN SEMI-SUPERVISADA DE LA INFORMACIÓN DE CONTEXTO

Raúl García Jiménez.
Tutor: Marcos Escudero Viñolo.
Ponente: Jesús Bescós Cano.

Julio 2016

DISEÑO Y DESARROLLO DE UNA HERRAMIENTA PARA LA EXTRACCIÓN SEMI-SUPERVISADA DE LA INFORMACIÓN DE CONTEXTO

Raúl García Jiménez
Tutor: Marcos Escudero Viñolo
Ponente: Jesús Bescós Cano

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2016

Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad del Gobierno de España bajo el proyecto TEC2014-53176-R (HAVideo) (2015-2017)

Resumen

Se ha desarrollado una herramienta de anotación de objetos de información contextual para zonas exteriores (carreteras,árboles, mar, etc.) y para zonas interiores (muebles, decoración, etc.). Las herramientas de anotación existentes están diseñadas para facilitar el etiquetado de personas o vehículos que interaccionan en la escena, pero no de los objetos con los que interaccionan. El diseño parte de una herramienta base del estado del arte. El objetivo principal es el etiquetado de un objeto a lo largo de un vídeo con la menor interacción posible del usuario pero manteniendo resultados de utilidad (relativa a la de la herramienta base). Para ello se aprovecha que los objetos contextuales varían poco en forma, color y posición durante el vídeo. En particular, se busca que la anotación realizada por el usuario en el primer cuadro del video se propague a lo largo del video. Además se diseñan e implementan estrategias de interacción en situaciones donde nuevos objetos entren en la escena. Podemos categorizar a la herramienta desarrollada como semi-supervisada.

El proceso comienza con la división de cada cuadro de la secuencia analizada en regiones homogéneas en color. Para ello se ha integrado un segmentador en regiones en la herramienta. En el primer cuadro (o cuando se activen los procesos de interacción) el usuario debe agrupar las regiones así obtenidas en regiones de interés con mayor entidad semántica (cercana a la definición de objetos) para adaptar el área espacial agrupada al contorno de los objetos contextuales que se desean anotar. Posteriormente, debe asignar una etiqueta de clase a cada región de interés agrupada. Estas anotaciones, en particular las regiones de interés sobre las que aplican, se utilizan como máscara en el siguiente cuadro para realizar la selección de regiones de interés automáticamente sin la interacción del usuario. De la misma forma se propagan las etiquetas de cada región de interés inicial a las regiones propagadas. La anotación así obtenida se propaga al siguiente cuadro del video mediante el mismo procedimiento, y así sucesivamente. Siguiendo esta sencilla estrategia, la herramienta diseñada tiene potencialmente la capacidad de reducir la interacción del usuario y a la vez, gracias al proceso de segmentación en regiones, de adaptarse a los cambios

graduales de forma, apariencia y posición de los objetos contextuales anotados.

Se han realizado dos pruebas experimentales. La primera evalúa el número de interacciones del usuario en el etiquetado de los objetos contextuales con cada una de las dos herramientas. La segunda evalúa la calidad de las anotaciones propagadas respecto a las anotadas por el usuario con la herramienta base. Los resultados de ambas evaluaciones validan el diseño y desarrollo de la herramienta realizada. Adicionalmente, se han realizado pruebas de experiencia subjetiva con usuarios. Para ello, se ha desarrollado un manual de instrucciones para la herramienta desarrollada y la herramienta base y un cuestionario para evaluar la experiencia de usuario. Las respuestas de 10 usuarios a este cuestionario sugieren que la herramienta diseñada facilita la anotación de los objetos contextuales en diferentes escenarios.

Palabras clave

segmentación, regiones, etiquetas, anotación, propagación, información contextual.

Abstract

A contextual annotation tool has been designed and developed. The tool is specially focused on the annotations of contextual outdoor (e.g. roads, trees or sea) and indoor areas (e.g. furniture or decoration, etc.). Existing tools are designed to ease the annotation of objects of interest in video-analysis applications, e.g.. people or cars, but do not considered the annotation of the contextual objects that surround them.

Proposed design builds on an existing annotation tool, which we name base tool. The base tool is modified in order to minimize the number of required user interactions while maintaining the degree of usability of the annotations in similar terms to those obtained via the base tool. To this aim, we take advantage of the intrinsic characteristic of contextual objects: gradual and moderate temporal variation in terms of appearance, shape and position. In general terms, the aim is to propagate an initial user annotation in the first frame along the whole video. User-tool interaction strategies are also included to cope with system failures and new object appearances. These interaction strategies define the designed tool as a semi-supervised annotation tool.

The process can be sketched as follows. First, each video frame is segmented in color homogeneous areas via a state of the art region segmentation method. Then, in the first frame and in every frame on which the interaction strategies are activated, the user has to group these regions into regions of interest. These regions on interest are entities of a higher semantic level (close to the object level). This process is devoted to adapt the spatial area to the contours of a particular contextual object. The so-obtained regions of interest are then user-labelled as members of a contextual class. Obtained annotations are used to propagate the information on the next frame by automatically grouping regions without the requirement of user interaction. The so-obtained annotations are subsequently propagated to the next video frame and so on until the end of the video. Through this naive strategy, the tool is potentially able to drastically reduce the number of required user interactions. Furthermore, due to

the region-driven scheme, is also able to adapt the annotations to gradual changes of shape, appearance and position of the annotated contextual objects.

In order to evaluate the goodness of the designed tool respect to the base tool, two different evaluations have been performed. The first one measures the number of user interactions required in the annotation of a set of sequences when using each of the tools. The second evaluates the quality of the propagated annotations respect to those obtained manually. Results of both experiments support the tool design and development. Finally, the quality of the user subjective experience is evaluated through a questionnaire. Responses of 10 users suggest that the designed tool ease the annotation of contextual objects in different and varied scenarios.

Keywords

segmentation, regions, labels, annotation, propagation, contextual information

Agradecimientos

En primer lugar agradecer a mi tutor, Marcos, por permitirme realizar este TFG y ayudarme en cada momento durante todo el año.

A los amigos que he hecho durante la carrera y los que empezaron conmigo, porque sólo ellos saben por lo que hemos pasado y por tantas horas de buenos momentos que han hecho que estos años sean inolvidables.

A mi chica Irene, por estar siempre a mi lado durante todos estos años. Tantas tardes de estudio aguantando mi estrés, compartiendo mis alegrías y también mis penas. Sin ti esto sería imposible.

En último lugar, pero no por ello menos importante, a mis padres, jamás podré agradecerles todo lo que hacen día a día por mí, brindándome su apoyo incondicional y confiando en mí en todo momento. Mil gracias por todo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En la actualidad, existe una gran demanda en el diseño, desarrollo y utilización de sistemas de monitorización automática en entornos de control y seguridad, lo que hace que se desarrollen continuamente aplicaciones basadas en el procesamiento de imagen y vídeo digital para su uso en vídeo-vigilancia. Las tareas comunes de los sistemas de video-vigilancia incluyen la detección y seguimiento de objetos de interés y la detección de los eventos (actividades e interacciones) realizados por estos objetos (activos) entre sí o con el entorno. Un campo interesante dentro de la vídeo-vigilancia, y mucho menos estudiado, es el del uso de la información de contexto. La información de contexto puede definirse como todos aquellos descriptores de los objetos no activos en un secuencia de video. Esta definición permite clasificar como información de contexto a evidencias tan dispares como: características de bajo nivel (color/textura), información sobre las condiciones de captura (calibración), descripciones semánticas de los objetos funcionales de la escena (uso y movilidad) y descripciones de la posición espacio-temporal y la representación en la imagen de los objetos contextuales. Diversos trabajos ([4]) han demostrado que el uso de la información de contexto de una escena y su propagación a lo largo de un vídeo resulta de utilidad para guiar, apoyar o constreñir etapas de análisis posteriores en un sistema de interpretación automática de vídeo.

Debido a la alta variabilidad existente entre los diferentes objetos que pueden catalogarse como contexto (variabilidad implícita en la definición realizada), las técnicas existentes para la extracción automática del contexto están limitadas a escenarios muy concretos. Como alternativa, existen mecanismos que permiten que el usuario suministre esta información (por medio generalmente de una interfaz gráfica

de usuario o GUI). Sin embargo la utilidad de las anotaciones proporcionadas por estas herramientas está limitada por uno de dos factores: i) asunción de estabilidad o ii) demanda de interacción contínua del usuario. En el primer caso, las anotaciones se realizan una vez y se asumen válidas para el resto del video, por tanto, su utilidad se reduce al análisis de secuencias grabadas por cámara estática y a objetos contextuales de baja (o nula) variabilidad en apariencia. En el segundo caso, se requiere que sea el usuario el que propague las anotaciones iniciales, convirtiendo la tarea en repetitiva y sujeta a errores de anotación. El trabajo descrito en este documento tiene como motivación principal la eliminación de ambas limitaciones.

1.2. Objetivos

El objetivo principal de este trabajo fin de grado es el diseño y desarrollo de una herramienta semi-supervisada para la generación automática de descripciones semánticas de los objetos funcionales de la escena, incluyendo: suelo, mesas, sillas, carreteras, aceras, barreras, pizarras, bancos, etc. Es decir, cualquier objeto de la escena con el que los objetivos activos del análisis, generalmente personas, pero también coches u otros vehículos, puedan interactuar directamente o indirectamente.

Para cumplir este objetivo principal se diseña un sistema mediante el cual:

- La información introducida por el usuario en el primer cuadro se propague adecuadamente al resto del vídeo.
- La interacción requerida por el usuario sea mínima, incrementando la eficiencia y usabilidad respecto a la herramienta base [1].
- Se disponga de mecanismos de interacción para corregir potenciales problemas de la propagación o situaciones no controladas.
- El ajuste espacial de los objetos anotados sea similar al obtenido manualmente.

El grado de consecución de estos objetivos será evaluado cuantitativa y cualitativamente mediante comparación con los resultados obtenidos haciendo uso de la herramienta base [1].

1.3. Organización de la memoria

La memoria se estructura en distintos capítulos:

- Capítulo 1. Se explica la motivación, el problema y el objetivo del proyecto.

- Capítulo 2. Aparece la herramienta de anotación manual contextual de la que se parte y otras herramientas de anotación semi-supervisadas que han servido como referencia.
- Capítulo 3. Vemos cada fase que lleva a cabo la herramienta.
- Capítulo 4. Explica las pruebas que se han realizado tanto por el alumno como por usuarios y los resultados que se han obtenido.
- Capítulo 5. Se desarrolla unas conclusiones de lo realizado y lo que se puede realizar en un futuro con la herramienta realizada.
- Bibliografía

Capítulo 2

Estado del arte

2.1. Introducción

Para la realización de este trabajo se han estudiado distintas herramientas de anotación de objetos a partir de las cuales se han extraído ideas que sirvan para desarrollar la herramienta propuesta.

En todos los casos, todas la herramientas usan una interfaz o GUI (Graphic User Interaction). Una GUI es una interfaz gráfica a partir de la cual interactúa el usuario con la herramienta. En este tipo de herramientas de anotación la interfaz suele mostrar las imágenes que componen el vídeo y una serie de botones con los que el usuario realiza los procedimientos en el vídeo.

En todas las herramientas el procedimiento general se basa en abrir un vídeo, del cual se muestra su primer cuadro. A partir de este cuadro, el usuario interacciona para generar regiones que definan a los objetos que desea anotar. Una vez se han generado las regiones, el usuario nuevamente interacciona para escoger una etiqueta que mejor describa al objeto que define cada región a partir de una ontología predefinida. Con este trabajo realizado en el primer cuadro, en cada herramienta se llevan a cabo unos procesos para que las regiones que definen a los objetos sigan a estos a lo largo del vídeo con la etiqueta asignada.

Lo habitual en las herramientas de anotación es el uso del *bounding box* como región que define a un objeto. El *bounding box* es el recuadro que contiene el objeto al que se pretende etiquetar. En casos como el de la anotación de personas o vehículos se pueden usar rectángulos como se ve en las herramientas estudiadas.

Figura 2.1: En la herramienta Vatic [5] se observa el uso de *bounding box* como regiones para definir a los vehículos y a las personas de la escena

En cambio, en la herramienta a desarrollar esta técnica no es válida, ya que regiones como carreteras o el cielo no se pueden definir adecuadamente (en general) mediante un rectángulo. Es en estas situacionesdonde la segmentación en regiones cobra vital importancia. La segmentación consiste en generar diferentes regiones en las cuales los píxeles que contienen guardan una relación entre si en color y/o brillo. De esta forma se consiguen regiones que definen objetos adaptándose a los bordes de éstos, compartiendo (al menos parcialmente) la misma forma que el objeto en cuestión.

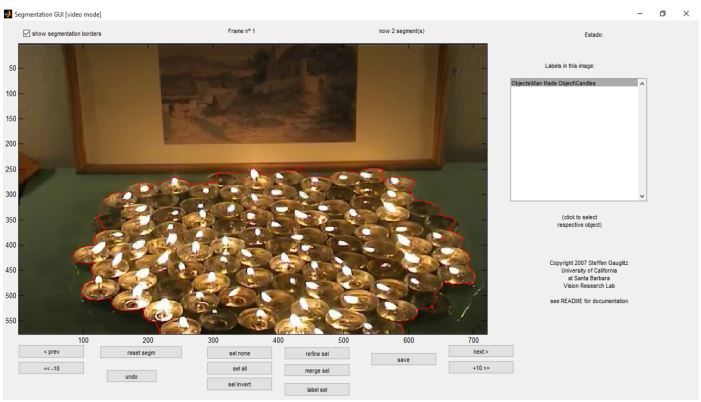


Figura 2.2: Esta es la interfaz de la herramienta base [1] de la que se parte para el desarrollo de la propuesta. En ella se observa la generación de regiones a partir de la segmentación. Estas regiones tienden a adaptarse parcialmente a los bordes de los objetos

Debido a que los objetos de interés no son por lo general completamente homogéneos, la segmentación en regiones tiene habitualmente el inconveniente de que se

generan más regiones de las deseadas (sobre-segmentación), por lo que en este caso se requiere de la intervención del usuario. En particular, el usuario interactúa con las regiones generadas por el segmentador, de forma que las regiones se adapten a los bordes y a la forma de los objetos a etiquetar.

A continuación se explorarán las distintas herramientas existentes que han sido estudiadadas, indicando las peculiaridades de cada una y las ideas que se han extraído de ellas.

2.2. Herramientas existentes

Las herramientas de anotación que se han estudiado son: Vatic [5], iVAT [6] y SVAS [7]. Además se ha estudiado la herramienta base [1] desde la que se parte para desarrollar la herramienta propuesta.

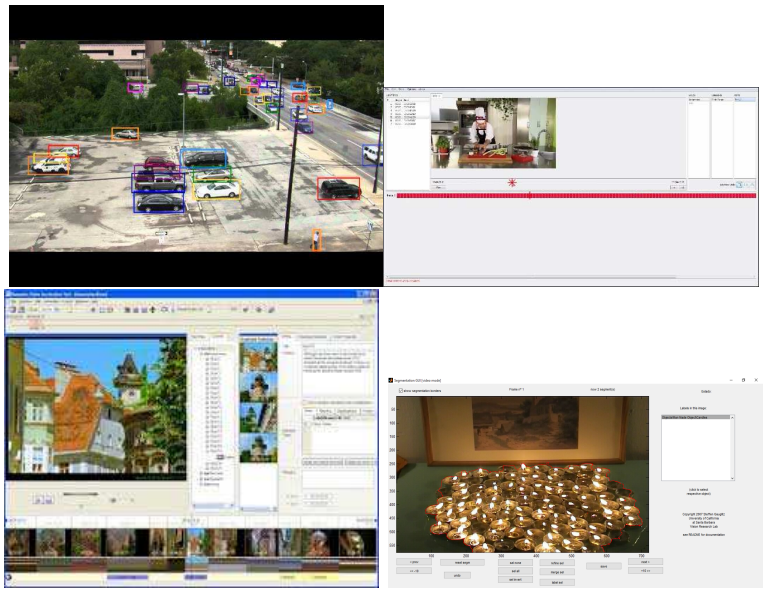


Figura 2.3: En cada imagen se muestra la interfaz de la herramienta de anotación en cuestión. 1ª fila, 1ª columna: Vatic [5]. 1ª fila, 2ª columna: iVAT [6] 2ª fila, 1ª columna: SVAS [7]. 2ª fila, 2ª columna: la herramienta base [1] de la que se parte

Vatic. La herramienta Vatic [5] tiene la desventaja de que para generar el camino que recorre un objeto de la escena, es necesario que el usuario interaccione en dos cuadros. El usuario genera el *bounding box* en un cuadro y posteriormente en un cuadro siguiente lo desplaza y lo adapta para que la herramienta genere el desplazamiento del *bounding box* entre ambos cuadros.

En la primera versión de la herramienta el desplazamiento entre cuadros del *boun-*

Herramienta	Tipo de anotación	Tipo de información a anotar	Número de interacciones mínimas (caso ideal)	Nivel de anotación
Vatic [5]	Semi-supervisada	Objetos que interaccionan	2 cuadros	*Bounding box*
iVAT [6]	Semi-supervisada	Objetos que interaccionan	1 cuadro	*Bounding box*
SVAS [7]	Semi-supervisada	Objetos que interaccionan	1 cuadro	*Bounding box*
Herramienta base [1]	Manual	Objetos que interaccionan/ Información contextual	Todos los cuadros	Regiones de interés
Herramienta propuesta	Semi-supervisada	Objetos que interaccionan/ Información contextual	1 cuadro	Regiones de interés

Tabla 2.1: Tabla resumen de herramientas de anotación

ding box, se propagaba mediante interpolación lineal entre los *bounding box* anotados, obteniendo buenos resultados para desplazamientos lineales. En versiones posteriores, los autores proponen un complejo sistema que basado en heurísticos extrapola la posición del objeto en los cuadros intermedios.

La herramienta propuesta no obliga al usuario a interactuar en dos cuadros distintos, reduciendo la intervención de éste. Además, con el uso de regiones en lugar del *bounding box* se puede anotar información contextual.

iVAT. Por otro lado, la herramienta iVAT [6] sí que permite al usuario interactuar en un único cuadro (en el caso óptimo sólo en el primero) y además modificar a lo largo del proceso los cuadros en los que el *bounding box* no se encuentra bien colocado. Esta herramienta diferencia entre dos sistemas integrados, el sistema de anotación (que toma del ViPER-GT) y el sistema de seguimiento. Mientras que en el sistema de anotación es el usuario quien realiza el procedimiento, en el sistema de seguimiento la herramienta actúa automáticamente para extrapolar los datos introducidos por el usuario a partir de un análisis que incluye segmentación, asociación y predicción de trayectorias.

Aunque esta herramienta permita al usuario intervenir en un único cuadro en el caso óptimo, el uso del *bounding box* impide anotar (en general) información contextual.

SVAS. Lo mismo sucede con la herramienta SVAS [7], que permite al usuario

interactuar en un único cuadro y posteriormente realizar modificaciones. Por su parte, esta herramienta realiza el proceso de desplazamiento del *bounding box* a través de puntos claves del vídeo, a partir de los cuales obtiene la posición del objeto en cada cuadro del vídeo.

Al igual que la herramienta anterior permite la intervención del usuario en un único cuadro en el caso óptimo, pero tampoco puede anotar información contextual por su uso del *bounding box*.

Herramienta base [1]**.** Por su parte, la herramienta base [1] de la que se parte en este TFG, es una herramienta en la cual el usuario debe interactuar en todos y cada uno de los cuadros del vídeo. Esto es una desventaja respecto a las otras herramientas, pero en contraposición, esta herramienta permite generar regiones que definan objetos como carreteras o vegetación a través de la segmentación. La herramienta usa la segmentación en regiones para definir objetos de la escena con la interacción del usuario.

Por su parte, la herramienta propuesta, también permite la anotación de ese tipo de objetos, pero además, en el caso óptimo, el usuario sólo interacciona en el primer cuadro.

2.3. Uso del contexto en el análisis

La información contextual consiste en el conjunto de objetos o circunstancias de una escena que son útiles para un análisis posterior. Esta informacion contextual tiene la peculiaridad de variar poco [8] tanto en color, como en la posición en la escena [9].

Los objetos que aportan información contextual se pueden dividir en dos grupos: en objetos funcionales [10] (como puertas o sillas) y áreas de influencia [11] (como carreteras o cielo). En ambos casos, a los objetos se les asignan descriptores que indican su estado y su posición.

Para realizar esta tarea de anotación de objetos existen actualmente distintas herramientas [4]. Un ejemplo es la codificación de mapas que definen los posibles caminos de los objetos en escenas 2D [12].

La información contextual varía con el paso del tiempo en la escena, por lo que debe actualizarse mediante la observación de los objetos de interés por el usuario. Por ejemplo, la ubicación de un objeto de información de contexto, es actualizada en [13] mediante indicadores del tráfico para detectar anomalías en la escena.

Por otro lado, la similitud entre dos instantes de la escena, es utilizada para hallar la apariencia de cada objeto entre medias con el uso del movimiento y del tracking

[14]. Algunos de los objetos de información contextual son dinámicos en la escena y hay que tenerlo en cuenta pues puede acarrear problemas como el de la extracción de fondos en fondos multimodales [15]. Este dinamismo de la escena se resuelve mediante seguimiento [16] o actualizando su módelo de apariencia.

La información contextual en una escena escalada es difícil de hallar desde un único punto de vista, por lo que se obtienen mejores resultados con distintas cámaras [17].

Por todo esto, el uso del contexto estático o dinámico en el reconocimiento del estado de una escena no es algo trivial o sencillo pero puede ser de gran utilidad para constreñir los procesos de análisis.

2.4. Conclusiones del estado del arte

A partir de lo que se ha visto, se llegan a una serie de conclusiones:

- Es deseable que, por un lado, la anotación la desarrolle el usuario y posteriormente la herramienta realice automáticamente el desplazamiento de las regiones a lo largo del vídeo.
- La herramienta a desarrollar debe permitir al usuario que interaccione únicamente en el primer frame en el mejor de los casos y que pueda modificar los cuadros posteriores si lo cree necesario.
- Para el seguimiento de los objetos a lo largo del vídeo se van a usar puntos clave que nos indican la posición de cada objeto en cada cuadro.
- Al no poder usarse *bounding box* por el tipo de objetos a etiquetar, se usará la segmentación en regiones para definir los objetos de la escena.

Capítulo 3

Sistema, diseño y desarrollo

3.1. Introducción

En este capítulo se describe cómo, partiendo de lo analizado en el estado del arte, se han desarrollado los nuevos módulos que se integran en la herramienta base [1] para conformar la actual. Se compara la herramienta base [1] y la actual a partir de los diagramas de bloques de las respectivas herramientas para facilitar la observación de las mejoras implementadas en la herramienta actual. Cada bloque o módulo del diagrama es explicado y detallado diferenciando entre los módulos que comparten ambas herramientas y los exclusivos de la herramienta actual, así como los problemas que surgen en para añadir cada módulo y las estrategias de integración necesarias para la creación de la herramienta actual.

3.2. Herramienta de anotación base

La herramamienta de la que se parte es una herramienta de anotación manual en la cual, el usuario debe modificar las regiones de una imagen para que definan un objeto u objetos a los que posteriormente se les asignan unas etiquetas de una lista preintroducida. Esta labor la debe hacer el usuario en cada cuadro del vídeo, ya que la herramienta no realiza ningún proceso automático entre cuadros, lo que conlleva a que la herramienta base [1] sea lenta y laboriosa.

3.2.1. Esquema y descripción general de la herramienta base

La estructura de esta herramienta base [1] puede observarse en la Figura 3.1.

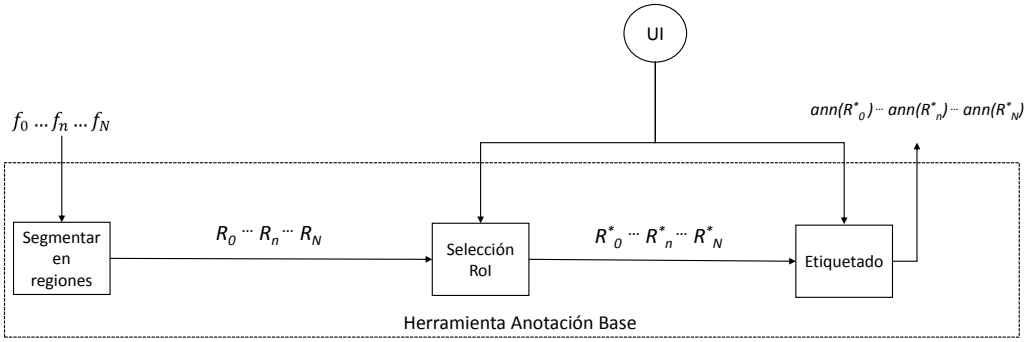


Figura 3.1: Herramienta base [1] en la cual cada uno de los cuadros del vídeo inicialmente es segmentado en regiones. Posteriormente el usuario interacciona y modifiqua las regiones dejando sólo aquellas de interés que definan un objeto. A continuación les asignará la etiqueta descriptiva que mejor corresponda, de modo que finalmente se obtenga el seguimiento de uno o varios objetos a lo largo del vídeo.

En la Figura 3.1:

- El subíndice n indica la posición del cuadro en el vídeo, siendo $n \in [1, N-1]$ y N el número de cuadros totales del vídeo.
- f_n es cada uno de los cuadros que entran en la herramienta.
- R_n representa a la imagen, de dos dimensiones (x e y), de regiones$\{\Omega^l_{1..j..J}\}_n$ que se extraen de la imagen inicial mediante el módulo segmentar (Sección 3.3.2.1).
- R^*_n es la imagen, de dos dimensiones (x e y), que contiene al conjunto de regiones de interés (RoI) que representan a los objetos contextuales anotados (Sección 3.3.2.2).
- $ann(R^*_n)$ representa la imagen, de dos dimensiones (x e y), de regiones de interés con las etiquetas que se les ha asignado a cada una de ellas (Sección 3.3.2.3).
- UI indica interacción del usuario (*User Interaction*) y aplica a los módulos en los cuales el usuario debe interactuar con la herramienta.

Como puede observarse en la herramienta base [1] de la que se parte, todos los cuadros del vídeo siguen el mismo camino en el cual se requiere la intereacción del usuario.

Sobre la herramienta base [1] original se implementó un módulo de lectura de video en un trabajo fin de grado anterior ([18]) además de funcionalidadas adicionales (como un detector facial que asignaba directamente un bounding box a las caras de

la imagen). En el desarrollo de este trabajo fin de grado se han descartado todas las funcionalidades adicionales a la lectura de video y a las incluidas en la herramienta base [1] original..

3.3. Herramienta de anotación propuesta

La herramienta de anotación propuesta al igual que la herramienta base [1], recibe los cuadros de un vídeo y los segmenta en regiones para que el usuario las agrupe en regiones de interés (RoI). Posteriormente, el usuario asigna una etiqueta a cada RoI en función del objeto contextual que contiene. La diferencia con la herramienta base [1] reside en que el usuario no tiene que repetir el proceso en cada cuadro, ya que la herramienta lo hace automáticamente a partir de un cuadro que ha procesado el usuario manualmente. De esta forma se reduce la labor del usuario que sólo interviene en el primer cuadro y posteriormente si debe realizar alguna modificación.

3.3.1. Esquema y descripción general

La estructura de esta herramienta base [1] puede observarse en la Figura 3.2, donde se utiliza la misma nomenclatura que para la herramienta base [1].

3.3.2. Funciones comunes a ambas herramientas

3.3.2.1. Segmentación en regiones

Cuando hablamos de segmentar nos referimos a dividir una imagen en determinadas regiones que definen los diferentes elementos de la escena, principalmente por la diferencias de color y brillo entre un elemento y otro. La imagen se divide completamente en regiones, es decir, ningún píxel queda sin asignarse. Las regiones no solapan entre sí, es decir, cada píxel sólo pertenece a una región.

El objetivo inicial de la herramienta propuesta es definir regiones de elementos contextuales tales como una carretera o el mar. Para esta labor no se pueden usar polígonos rectangulares ya que no definirían correctamente los elementos contextuales. En cambio, la segmentación en regiones genera una partición de la imagen en regiones diferenciadas cuyos bordes tienden a ajustarse a los bordes de los objetos.

La segmentación se realiza siguiendo un esquema *offline* al comienzo del proceso de análisis, es decir, toma como entrada todos los cuadros del vídeo ($f_{0...N}$) y retorna las imágenes de regiones ($R_{0...N}(x, y)$) y, opcionalmente, la representación gráfica

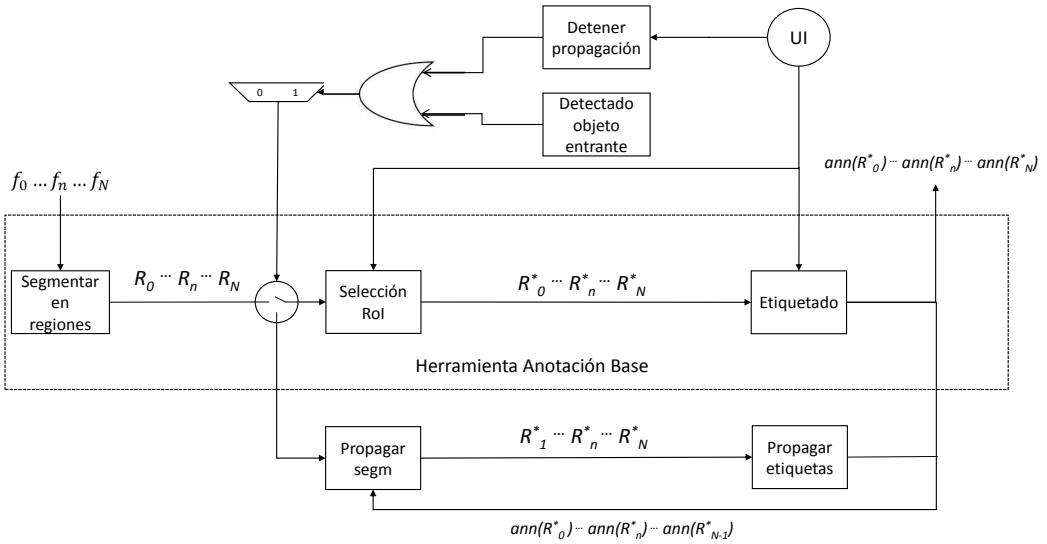


Figura 3.2: Herramienta actual, en ella los cuadros de un vídeo son segmentados al inicio. Salvo el caso del primer cuadro que obligatoriamente el usaurio debe seleccionar las regiones que definen objetos de interés y asignarles etiquetas, el resto de cuadros pueden recorrer otro camino. Por el otro camino la herramienta realiza una propagación homóloga a la selección de regiones y a la asignación de etiquetas en la cual el usuario no interviene. El usuario puede decidir parar el proceso o la herramienta lo hace si detecta un nuevo objeto en la escena, en este caso, el usuario debe volver a realizar el proceso manualmente en ese cuadro. Finalmente se obtiene, al igual que en la herramienta base [1], el seguimiento de los objetos de interés a lo largo del vídeo.

donde a cada región se le asigna la moda del color de los píxeles asociados a la región ($M_{0\ldots N}(x, y)$). En las imágenes de regiones, cada píxel se representa por una etiqueta que define implíctamente el conjunto de regiones mediante la agrupación en conjuntos de píxeles: $\{\Omega^l_{1..j..J}\}_n$

La segmentación en regiones en un proceso totalmente automático que no requiere de interacción del usuario. Sin embargo, las regiones obtenidas restringen la anotación de usuario (que se realiza agrupando regiones en regiones de interés como se decribirá en la Sección 3.3.2.2). En particular, los bordes no detectados en la partición en regiones no podrán recuperarse en la anotación de los objetos. Denominaremos a este problema sub-segmentación. Por otro lado la división de un objeto en múltiples regiones hace que el proceso de agrupación y etiquetado posterior sea tedioso, pues estas regiones deberán ser fusionadas por el usuario. Denominaremos a este problema sobre-segmentación. Podemos ver un ejemplo en la Figura 3.3,

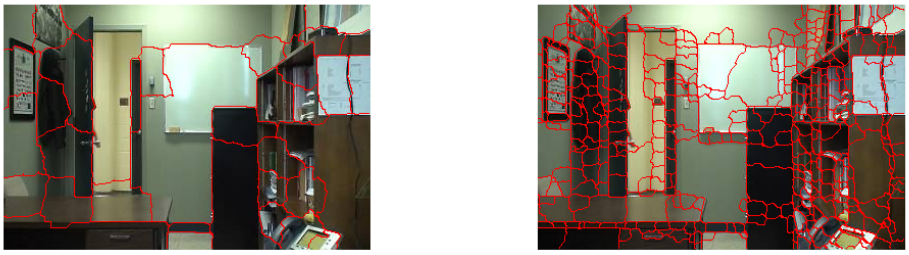


Figura 3.3: Imagen de la izquierda (Sub-segmentación): Imagen con una segmentación muy gruesa en la cual existen bordes de los objetos que no han sido detectados. Imagen de la derecha (Sobre-segmentación): Imagen con una segmentación muy fina en la cual se generan más regiones de las que interesar para la anotación

Para minimizar ambos problemas, la herramienta base [1] realiza una segmentación con varios niveles de agrupación, donde las regiones del nivel $l+1$ son particiones de las del nivel l. En concreto se generan $L = 5$ niveles o particiones en regiones por cada cuadro; siendo el primer nivel el que menos regiones tiene y el quinto y último nivel el que más regiones tiene. Nótese que en cada nivel se añaden, pero no se eliminan, bordes respecto al nivel anterior. De esta forma en el siguiente bloque de anotación el usuario puede variar entre los distintos niveles de segmentación hasta encontrar una segmentación óptima.

Una imagen de regiones a un determinado nivel (R_n^l) define un conjunto de regiones $\{\Omega_{1..j..J}^l\}_n$, donde la primera región del conjunto vendrá identificada como $\min(R_n^l) = 1$ y la última como $\max(R_n^l) = J$.

El segmentador inicial que se usaba en la herramienta base [1] es el segmentador de Felzenszwalb [3] Experimentalmente, observamos que el segmentador tenía problemas en algunos casos, a saber:

- Las regiones generadas no se adaptaban con precisión a los bordes de los objetos de cada cuadro.
- En ocasiones, no se separaban en diferentes regiones dos objetos distintos de la escena.
- Al llegar al nivel $l = L$ se generaban regiones demasiado pequeñas que eran difíciles de manejar por el usuario.

Como alternativa, se propone usar un segmentador en superpíxeles [2]. Los superpíxeles tienden a agrupar píxeles conexos con el mismo color y brillo generando regiones

menos irregulares y más ajustados a los contornos de los objetos. En la tabla 3.1 pueden observarse la segmentación en regiones de una imagen con el segmentador de Felzenszwalb [3] y el de superpíxels [2], ambos en el nivel $l = 1$ (nivel con menos regiones) y en $l = L$ (nivel con más regiones).

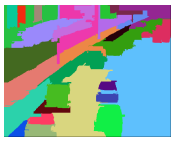
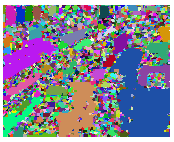
Imagen a segmentar 	
Imagen de 1^{er} nivel de Felzenszwalb  	Imagen de 5^{er} nivel de Felzenszwalb  
Imagen de 1^{er} nivel de superpíxels  	Imagen de 5^{er} nivel de superpíxels  

Tabla 3.1: Siendo la imagen superior la imagen a segmentar f_n: en la 2ª fila, 1ª columna se encuentra la imagen de etiquetas R_n^1 resultado de la segmentación Felzenszwalb [3] para el nivel $l = 1$(izquierda) y la imagen de modas M_n^1 para el mismo nivel (derecha). En la 2ª fila, 2ª columna se encuentra la imagen de etiquetas R_n^5 resultado de la segmentación Felzenszwalb [3] para el nivel $l = 5$(izquierda) y la imagen de modas M_n^5 para el mismo nivel (derecha). En la 3ª fila y 1ª columna se encuentra la imagen de etiquetas R_n^1 resultado de la segmentación superpíxels [2] de nivel $l = 1$(izquierda) y la imagen de modas M_n^1 para el mismo nivel (derecha). Finalmente, en la 3ª fila, 2ª columna se encuentra la imagen de etiquetas R_n^5 resultado de la segmentación superpíxels [2] de nivel $l = 5$(izquierda) y la imagen de modas M_n^5 para el mismo nivel (derecha). Las imágenes de regiones (R_n), cada región tiene un valor específico en todos los píxeles que contiene. Como se puede observar, para ambos segmentadores a más nivel se generan más regiones en la imagen. Por otro lado, la imagen de modas M_n, cada región se muestra con los colores RGB medios de todos los píxeles que contiene

3.3.2.2. Selección de regiones de interés

La selección de regiones de interés es la labor que realiza el usuario a partir de la imagen de regiones obtenida y sus cinco niveles, en la cual dividiendo o uniendo

regiones genera finalmente una imagen de regiones de interés que identifiquen las secciones de la escena que son susceptibles de ser etiquetadas posteriormente.

Para realizar esta labor, el usuario debe utilizar las funcionalidades implementadas en la interfaz. El usuario segmenta en más regiones una región seleccionada pasando al siguiente nivel de los cinco de la imagen de regiones. Por otro lado, une regiones seleccionadas en una sola, proporcionadno el mismo valor en cada punto de las regiones seleccionadas. Finalmente con esta interacción del usuario, del conjunto de imágenes de regiones obtenida ($\{R_n^l, l = 1...L\}$) se obtiene una única imagen de regiones de interés (R^*).

Un ejemplo de selección de regiones de interés de una imagen de regiones es la siguiente Figura 3.4, en la que el usuario ha seleccionado como regiones de interés dos coches en la escena del ejemplo anterior.

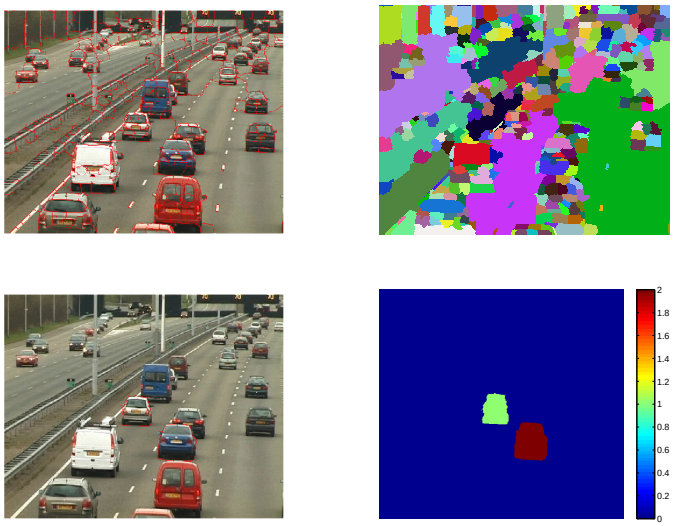


Figura 3.4: Selección de regiones de interés. En la 1ª fila, 1ª columna se encuentra la imagen en regiones tal y como se muestra en la interfaz. En la 1ª fila, 2ª columna, se encuentra la imagen en regiones (R_n) donde se puede observar que cada región tiene un identificador diferente. En la 2ª fila, 1ª columna, se muestra la imagen con las regiones de interés seleccionadas por el usuario. En la 2ª fila, 2ª columna se observa la imagen de regiones de interés (R_n^*) en la cual cada región tiene un identificador, pero sólo hay tres regiones (dos coches y fondo)

3.3.2.3. Etiquetado

Con la imagen de regiones de interés seleccionadas por el usuario se pasa a la fase de anotación de las regiones de interés. Al etiquetar se asigna a la región seleccionada

una descripción sobre el objeto que define. Nuevamente toma parte el usuario, que debe seleccionar la región de interés a etiquetar y seleccionar la clase de objeto más conveniente.

La etiqueta debe ser previamente definida en un fichero de texto. En este fichero las etiquetas se agrupan en categorías creando una ontología de objetos muy sencilla. Por ejemplo, para el ejemplo representado en la Figura 3.4 las etiquetas asociadas por el usuario a las regiones de interés serían:

1. Objects/Transportation/Car
2. Objects/Transportation/Car1

En las etiquetas se distingue una ontología que aportan información al objeto anotado.

Como se puede observar una de las dos regiones lleva un número al final de la etiqueta, por lo que no se repite ninguna etiqueta en el mismo cuadro. Esto es una mejora implementada para que se diferencie cada etiqueta en cuadros posteriores.

Una vez asignada la etiqueta que más le conviene a cada región, se genera la imagen con un conjunto de regiones de interés con una o varias etiquetas asignadas a sus respectivas regiones ($ann(R^*_{0...N})$).

3.3.2.4. Salida

Finalmente al guardar se almacenan en distintos ficheros los datos introducidos, tanto las regiones de la imagen como las etiquetas asignadas. Esta es la salida suministrada para cada cuadro por la herramienta base [1]. Por el contrario, en la herramienta actual esta salida se usa para la propagación, ya que se usa esta imagen resultado del cuadro anterior para crear la del cuadro actual y así sucesivamente, Los procesos de propagación se describirán en las siguientes secciones 3.3.3.1 y 3.3.3.2.

3.3.3. Funciones exclusivas de la herramienta actual

En la herramienta actual existe la opción de propagar la información realizada en un cuadro a los siguientes, por lo que con la imagen resultado del cuadro inicial $ann(R^*_0)$ se pueden generar el resto de imágenes resultado $ann(R^*_{1...N})$ en el caso óptimo en el que el usuario sólo intervenga en el primer cuadro.

3.3.3.1. Propagar segmentación

El objetivo de este bloque es utilizar la anotación anterior para realizar la función homóloga al bloque de selección de regiones de interés (Sección 3.3.2.2), sin necesidad de intervención del usuario.

Este módulo recibe la salida del resultado del cuadro anterior de la que lee la imagen de regiones de interés (R^*_{n-1}) a partir de la cual se quiere obtener la imagen de regiones de interés del cuadro actual (R^*_n).

Para este proceso se parte de la imagen de segmentación de segundo nivel (R^2_n) obtenida en el módulo segmentar (Sección 3.3.2.1). Se ha establecido la imagen de segundo nivel (R^2_n) como referencia para el proceso ya que es la que mejor resultado aporta. Con el primer nivel, al ser una segmentación más gruesa con regiones más grandes, puede que no se adapten todas las regiones a los bordes del objeto. Por otro lado, los niveles superiores al tener muchas regiones pequeñas, realiza un proceso más costoso y lento.

El proceso que se realiza consiste en utilizar las regiones de interés de la imagen del cuadro anterior (R^*_{n-1}) como máscara. Se solapa esta máscara en la misma posición de la imagen de regiones del cuadro actual (R^2_n).

Para ello, sea $mk^*(x,y)_j$ la máscara de la región de interés j' del cuadro anterior $\{\Omega_j\}^*_{n-1}$, construida por activación binaria de todos los píxeles pertenecientes a la región de interés j' en la imagen de regiones de interés R^*_{n-1}. Sea $mk(x,y)_{j'}$la máscara de la región j' en la imagen de regiones en el cuadro actual R_n, el solape $\sigma(j')$ de la máscara $mk(x,y)_{j'}$ de cada región j' en el cuadro actual con $mk^*(x,y)_j$se calcula como:

$$\sigma(j') = \left|mk^*(x,y)_j \cap mk(x,y)_{j'}\right| \tag{3.1}$$

, donde $|mk|$ devuelve el número de píxeles distinto de 0.

Finalmete se detectan como regiones de interés, a todas aquellas regiones cuya máscara solape al menos un 50 % de sus píxeles con alguna de las máscaras de la regiones de interés del cuadro anterior:

$$\{\Omega_j\}^*_n \leftarrow \Omega_{j'}, \forall j' \text{ tal que } \frac{\sigma(j')}{mk(x,y)_{j'}} > 0{,}5 \tag{3.2}$$

De esta forma se generan un conjunto de regiones deinterés homólogo al del cuadro anterior, un ejemplo gráfico de este proceso puede observarse en la Figura 3.5.

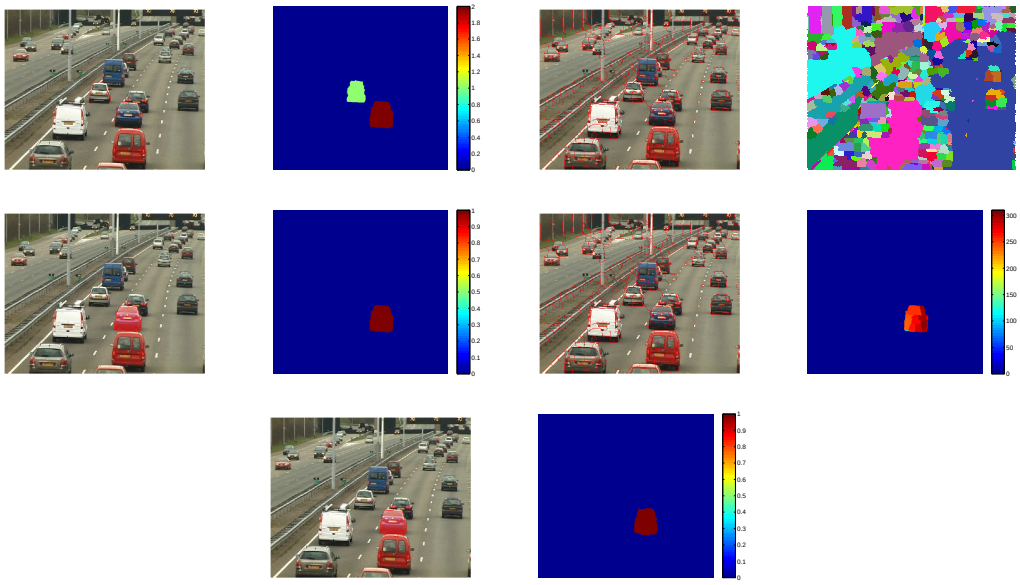


Figura 3.5: La imagen de la 1ª fila, 1ª columna es la imagen de regiones de interés del cuadro anterior (R^*_{n-1}) que se usa como referencia. La imagen de la 1ª fila, 2ª columna es la representación de la anterior en los valores de cada región. La imagen de la 1ª fila, 3ª columna es la imagen de regiones del cuadro actual (R_n) a modificar. La imagen de la 1ª fila, 4ª columna es la representación de los identificadores de las regiones de la imagen anterior. Como se puede observar estas imagénes guardan cierta relación con la Figura 3.4 en la que se veía como se optimizaba la segmentación con la intervención del usuario. Y es que este bloque realiza la función homóloga al bloque de selección de regiones de interés (Sección 3.3.2.2) pero sin intervención del usuario. Posteriormente en la 2ª fila, 1ª columna se muestra la imagen de una región del cuadro anterior ($\{\Omega_j\}^*_{n-1}$) que se va a propagar y en la 2ª columna la región como máscara $mk^*(x,y)_j$. La 2ª fila, 3ª columna se muestra la imagen de regiones del cuadro actual (R_n). Por su parte, la imagen de la 2ª fila, 4ª columna muestra las regiones del cuadro actual solapadas con la máscara ($\sigma(j')$). Finalmente, en la última fila se observan la imagen de regiones de interés del cuadro actual (R^*_n), resultado del proceso y la imagen con los valores de las regiones de interés.

Con este nuevo módulo de la herramienta actual se consigue que el usuario no tenga que volver a realizar el módulo de selección de regiones de interés (Sección 3.3.2.2) en cada cuadro salvo si detecta algún fallo y debe intervenir (Sección 3.3.3.4). Por tanto, se incrementa la eficiencia del proceso y se disminuye el esfuerzo del usuario.

3.3.3.2. Propagar etiquetas

Al igual que en el módulo anterior, se busca propagar información que ya se ha establecido por el usuario sin que éste tenga que volver a intervenir. En este caso se busca asignar las mismas etiquetas que tenían las regiones de interés etiquetadas en el cuadro anterior que equivalen a sus regiones de interés homólogas del cuadro actual. Es decir el objetivo es construir $ann(R^*_n)$ a partir de la información contenida en $ann(R^*_{n-1})$. Por lo tanto este módulo busca generar una salida equivalente al módulo descrito en la Sección 3.3.2.3 pero nuevamente sin intervención del usuario.

Con este fin, se propone un esquema para poder asociar anotaciones entre dos cuadros consecutivos $n-1$ y n.

Para ello la herramienta hace uso de un único punto de cada región de interés etiquetada en $ann(R^*_{n-1})$ que, asumiendo desplazamientos de la cámara no muy grandes, tiende a estar contenido también en la región de interés homóloga del cuadro siguiente. Esta asunción es generalmente válida en los objetos contextuales, que cubren grandes áreas de la imagen.

Como no se realiza ninguna predicción del movimiento de la cámara, una estrategia válida para la selección del punto es la extracción del centro geodésico Cg_j de una región de interés anotada $j \in \Omega^*_{j,n-1}$, siendo $\{\Omega_{1..j..J}\}^*_{n-1}$ el set de regiones de interés definido por R^*_{n-1}.

El centro geodésico es el punto más interno de j, es decir, el más alejado de los contornos de la región de interés. Por lo tanto, aunque el objeto definido por esa región se mueva sustancialmente en la escena, es de esperar que una hipótetica región de interés homóloga en el siguiente cuadro también contenga ese punto.

El centro geodésico puede definirse como un punto de la imagen asociado a la región de interés:

$$Cg_j = (x, y); \quad x \in [1, H], \quad y \in [1, W], (x, y) \in j \tag{3.3}$$

, siendo H la altura y W el ancho de la imagen.

Para hallar el centro geodésico se comienza por la imagen de regiones de interés (R^*_{n-1}) y se crea una máscara binaria de la región de interés j: $mk(x, y)_j$. Sobre esta máscara se extrae la transformada de distancias $D(x, y)_j$ en la cual cada punto de la transformada de distancias pasa a indicar la distancia desde ese punto al punto mínimo (0) más cercano en la máscara $mk(x, y)_j$. Cg_j se obtiene como el punto con mayor distancia al exterior de la región:

$$Cg_j = \underset{(x,y)}{argmax}(D(x, y)_j) \tag{3.4}$$

Finalmente, la etiqueta de j se propaga a j' ($ann(j') \leftarrow ann(j)$), donde j' es la región en la imagen de regiones de interés R_n^* que contiene Cg_j.

Repitiendo este proceso para todas las regiones de interés definidas en R_{n-1}^* se etiqueta la imagen de regiones de interés en el cuadro actual (R_n^*) obteniendo $ann(R_n^*)$ sin que el usuario haya interactuado con la interfaz. Así se obtiene a la salida un resultado similar al de la herramienta base [1], con la diferencia de que el usuario no ha intervenido en el procesamiento de cada cuadro. Salvo el primer cuadro y aquellos en los que el usuario haya decidido intervenir voluntariamente.

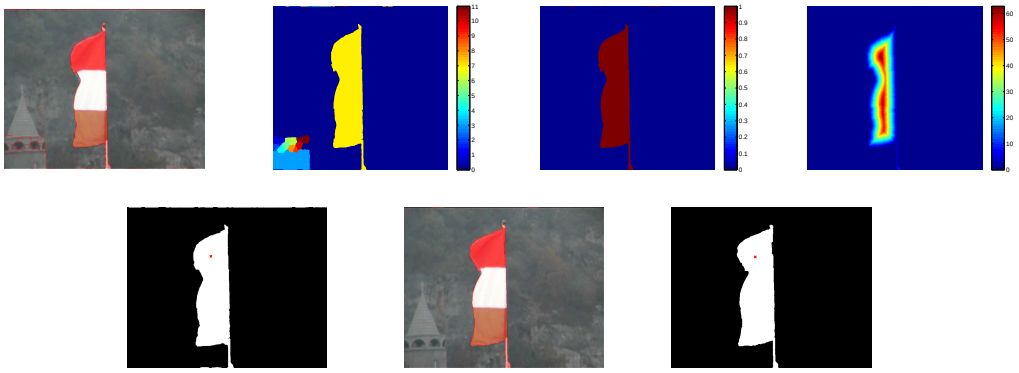


Figura 3.6: La imagen de la 1ª fila, 1ª columna muestra f_{n-1} con los contornos de la región de interés j superpuesto en rojo. En la 1ª fila, 2ª columna se muestra la imagen de regiones de interés (R_{n-1}^*). En la 1ª fila, 3ª columna, la máscara de la región a propagar ($mk(j)$). En la 1ª fila, 4ª columna se muestra la transformada de distancias de la región($D(x,y)_j$). En la 2ª fila, 1ª columna se muestra una imagen del cuadro anterior con la región j en blanco y su centro geodésico (Cg_j) marcado con una equis roja. En la 2ª fila, 2ª columna se muestra $f_n$con los contornos de la región homóloga a j superpuestos en rojo. Finalmente, en la 2ª fila, 3ª columna se representa la imagen del cuadro actual con la región homóloga j en blanco y el mismo centro geodésico (Cg_j) hallado anteriormente con una equis roja. Como se puede observar, el centro geodésico (Cg_j) hallado en la región j del cuadro anterior, está también contenido en la región homóloga del cuadro actual.

3.3.3.3. Detector objetos entrantes

El objetivo de este módulo es el de detener el proceso de propagación de la herramienta por un nuevo objeto. Este nuevo objeto entra en la escena y puede que sea necesario anotarlo o que modifique la forrma de una región de interés anotada.

Al detener la propagación el usuario vuelve a la Sección de selección de regiones de interés (Sección 3.3.2.2) para realizar las modificaciones que crea necesarias.

Para detectar un objeto entrante en la escena, la herramienta parte de la imagen

del cuadro actual (f_n) y la del cuadro siguiente (f_{n+1}). Se calcula el flujo óptico entre ambas imágenes mediante [19]. Con esto obtenemos componentes horizontal $V_x(x, y)$ y vertical $V_y(x, y)$ del movimiento seguido por cada píxel (x, y).

Área de interés La herramienta sólo detecta el flujo óptico en los bordes de la imágen, ya que sólo interesa las zonas en las que entran los nuevos objetos a la escena. La herramienta calcula el flujo óptico en los 50 primeros puntos de los cuatro bordes de la escena.

Técnica de detección A partir de los vectores de movimiento se halla el módulo de desplazamiento:

$$Mod(x, y)_n = \sqrt{V_x(x, y)^2 + V_y(x, y)^2} \tag{3.5}$$

En este módulo de desplazamiento, cada punto tiene como valor la cantidad que se ha movido de un cuadro a otro. En la herramienta lo que interesa es el desplazamiento del contorno del objeto, ya que interesa detectar el objeto entrante una vez, cuando empieza a aparecer en la escena, y no en cada cuadro en el que el objeto sigue entrando en la escena.

Por esta razón se halla el gradiente del módulo, a partir del módulo del cuadro actual con el siguiente y el del cuadro anterior con el actual:

$$Gmod(x, y)_n = \sqrt{(Mod(x, y)_n - Mod(x, y)_{n-1})^2} \tag{3.6}$$

Con el gradiente del módulo se obtiene un valor que indique el movimiento en el área de interés. Cuanto más alto es el valor del píxel en el gradiente del módulo mayor movimiento habrá, por lo que se halla un valor de movimiento del cuadro:

$$Mov_n = \sum_{x=1}^{H} \sum_{y=1}^{W} Gmod(x, y)_n \tag{3.7}$$

A partir de este valor, la herramienta lo compara con el obtenido para el cuadro anterior (Mov_{n-1}) y compara si ha aumentado sustancialmente respecto a él. Si es así, considera que hay un objeto entrando y detiene la propagación, avisando al usuario de que un nuevo objeto está entrando en la escena.

Umbral de detección El umbral que debe superar este valor es dinámico, ya que en determinadas escenas (como escenas con agua) lo normal es que haya algo de

movimiento, pero no tiene porque indicar que entre un nuevo objeto, por lo que el umbral depende del valor anterior (Mov_{n-1}).

Para inicializar el umbral en el primer cuadro, como se puede suponer, es un caso especial. Al hallar el valor de movimiento del primer cuadro frente al segundo (Mov_1) no se puede comparar con uno anterior. Por ese motivo, en este caso, la herramienta halla los valores de movimiento del primer al segundo cuadro (Mov_1), del segundo al tercero (Mov_2) y del tercero al cuarto (Mov_3) y toma el mínimo de esos valores como referencia al movimiento que hay en la escena para usarlo como umbral (Mov_0).

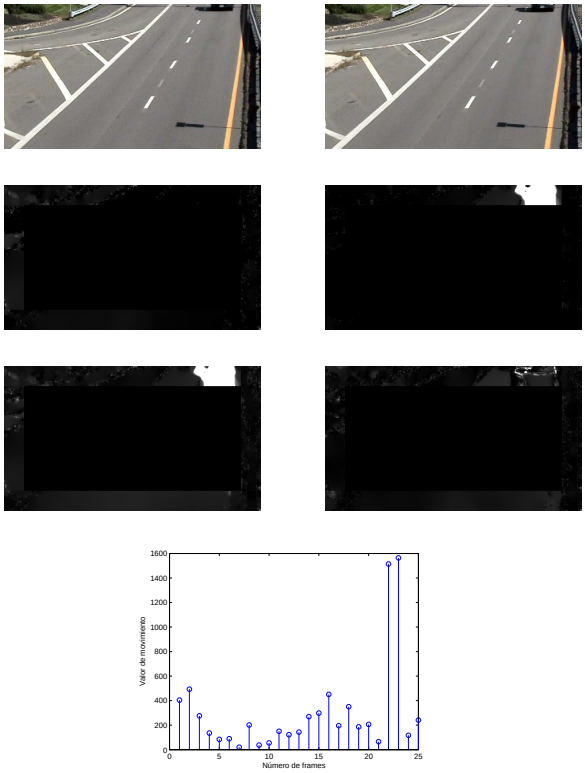


Figura 3.7: En la 1ª fila se muestran un cuadro del vídeo y el siguiente entre los que se va a detectar movimiento. En la 2ª fila, 1ª columna se muestra la imagen de los vectores de movimiento en el eje horizontal (V_x). En la 2ª columna se muestra respecto al eje vertical (V_y). En la 3ª fila, 1ª columna se muestra el módulo de los vectores de movimiento ($Mod(x,y)_n$), donde cada punto contiene la cantidad de movimiento de un cuadro a otro. En la 3ª fila, 2ª columna aparece la imagen del gradiente del módulo ($Gmod(x,y)_n$) en el cual el movimiento de un objeto sólo se muestra en su contorno y no en el interior. La imagen de la última fila corresponde al valor de movimiento en una escena a lo largo de sus cuadros, donde en los picos significa que hay alto movimiento, por lo que en estos casos será donde la herramienta indique objeto entrante

3.3.3.4. Detener propagación

Este módulo permite detener la propagación en cualquier momento por el usuario. En los casos en los que la propagación de la segmentación o de las etiquetas no es correcta, el usuario puede intervenir para detener la propagación y realizar las modificaciones necesarias para volver a propagar esta vez correctamente. Equivaldría a volver al camino que comparte con la herramienta base [1] (Secciones 3.3.2.2 y 3.3.2.3), pero simplemente en ese cuadro, ya que una vez procesado el cuadro por el usuario, vuelve a realizar la propagación desde ese cuadro.

Esto proporciona al usuario el control de la propagación, ya que si algo piensa que no es correcto puede arreglarlo durante la propagación y no a posteriori.

3.4. Estrategia de integración

Durante la implementación de los nuevos módulos han surgido una serie de problemas. Por este motivo en varios de estos módulos se siguen estrategias de integración que solucionan estos problemas y son detallados en esta Sección.

3.4.1. Integración en segmentar

En el módulo de segmentar en regiones (Sección 3.3.2.1), las imágenes de segmentación que se obtienen por defecto con el segmentador en superpíxeles [2] tienen un problema para integrarlas en la herramienta. Las regiones están separadas por líneas de valor cero y la herramienta considera como región todos aquellos píxeles de igual valor, lo que provocaría que la herramienta considerara todos los píxeles con valor cero como una región más. Es por ello que se obligan a estos píxeles a tomar el valor del píxel no nulo más cercano. De esta forma se eliminan los bordes y se obtiene la imagen de regiones definitiva (R_n).

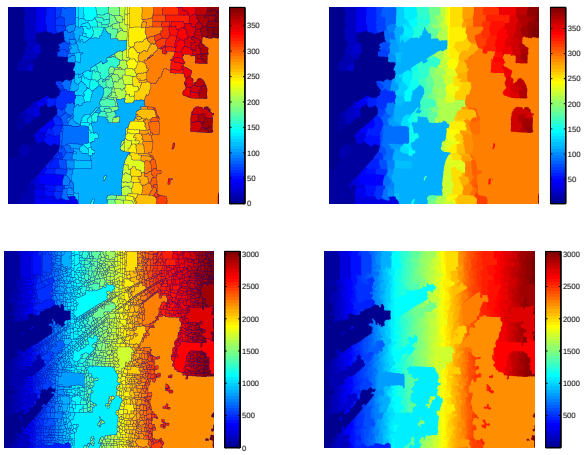


Figura 3.8: En primer lugar se observa el caso de la imagen de primer nivel (R_n^1) en la cual cada borde tiene el valor cero en todos sus puntos. Se asigna a los píxeles de valor cero, el valor no nulo más cercano. Así se obteniene la imagen de la derecha, en la cual no existe el valor cero como se observa. Abajo es el mismo caso pero esta vez con el quinto nivel (R_n^5)

3.4.2. Integración en propagar segmentación

Cuando se va a propagar la segmentación (Sección 3.3.3.1) del cuadro anterior (R_{n-1}^*) a la actual (R_n), se trata previamente la imagen de segmentación actual. Aunque puede que a simple vista no se aprecie, hay regiones separadas espacialmente con el mismo valor y por lo tanto consideradas de la misma región. Esto puede llevar a problemas durante la propagación ya que puede que esta separación espacial aumente. Para ello se le asigna un nuevo valor a una de las dos partes separadas de la misma región.

Tras este proceso la imagen de segmentación del cuadro actual (R_n) tiene todas sus regiones con el mismo valor unidas espacialmente. Y es con esta imagen de segmentación (R_n) con la que se realiza la propagación.

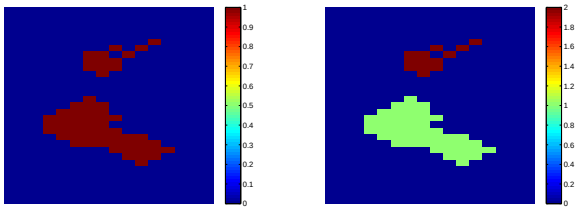


Figura 3.9: En este ejemplo la imagen de la izquierda contiene dos regiones separadas espacialmente que al tener ambas regiones el mismo valor identificativo, la herramienta la considera una misma región. Al separarlas se obtiene la imagen de la derecha, en la cual las regiones separadas tienen un valor distinto, lo que provoca que la herramienta las considere dos regiones ajenas la una a la otra

3.4.2.1. Integración en propagar etiquetas

Durante la propagación de etiquetas (Sección 3.3.3.2), puede darse el caso de que en el siguiente cuadro salga de la escena el objeto etiquetado. Por esta razón, al asignar la etiqueta correspondiente en el punto geodésico indicado, se asignaría la etiqueta a un objeto erróneo, por lo que además del centro geodésico se halla el tamaño en número de puntos de cada región etiquetada.

$$T(\Omega)_j = \sum_{x=1}^{H} \sum_{y=1}^{W} mk(x,y)_j \tag{3.8}$$

Si durante la propagación se detecta que el tamaño de la región ($T(\Omega)_j$) ha variado mucho respecto a la región homóloga del cuadro anterior ($T(\Omega)_{j-1}$), significa que el objeto al que identificaba la región ha abandonado la escena, por lo que directamente no se halla el centro geodésico (Cg_j) y no se propaga la etiqueta.

Capítulo 4

Experimentos Realizados y Resultados

4.1. Descripción del entorno gráfico de la herramienta de anotación

En este apartado se ve el diseño de la interfaz de la que se partía y cómo está la actual viendo cada parte del entorno gráfico.

4.1.1. Herramienta base

La herramienta base [1], como ya se ha comentado, no permitía la propagación de información de un cuadro a otro por lo que hay que ir interactuando de cuadro en cuadro con la ayuda de la interfaz de la Figura 4.1.

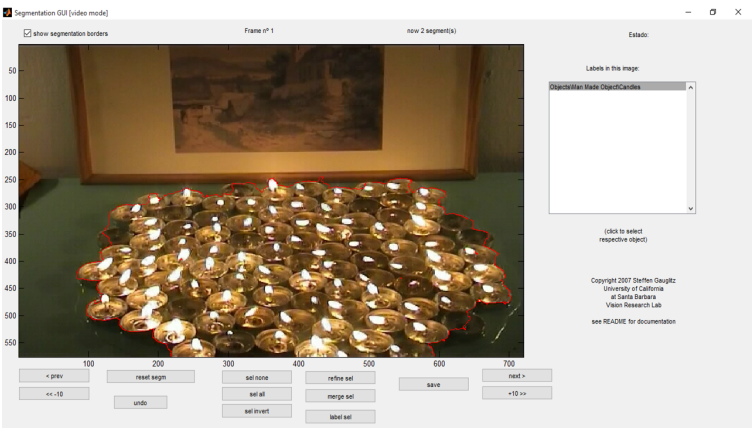


Figura 4.1: Interfaz de la herramienta base [1]

En esta interfaz se ven distintos botones con los que interactúa el usuario:

- Los botones *next* y *prev* con los que el usuario se desplaza hacia el cuadro siguiente o el anterior respectivamente y por otro lado $+10$ y -10 con los que se desplaza también pero de diez en diez cuadros.
- El botón *reset segmention* carga la segmentación inicial de primer nivel (R^1) de nuevo para volver a empezar con la selección de regiones de interés.
- El botón *undo* vuelve al estado justo anterior de la segmentación, pero sólo una vez, no vuelve a estados anteriores a ese.
- Los botones *sel all*, *sel none* y *sel invert* ayudan al usuario en la fase de selección de regiones de interés. El botón *sel all* selecciona todas las regiones del cuadro, al contrario que *sel none*, que deselecciona las regiones ya seleccionadas. Por su parte el botón *sel invert* selecciona las regiones no seleccionadas y deselecciona las seleccionadas.
- Los botones $refine\ sel$ y $merge\ sel$ son los más importantes pues se encargan de dividir las regiones seleccionadas o de unirlas respectivamente. Con el uso de estos botones, el usuario, convierte la imagen de regiones (R) en la imagen de regiones de interés (R^*).
- El botón *label sel* se encarga de abrir un desplegable con las etiquetas disponibles para que el usuario seleccione la que le va a asignar a la región seleccionada y de esta forma etiquetar el cuadro y generar las etiquetas de las regiones de interés ($ann(R^*)$).
- El botón *save* guarda el estado de la segmentación y de las etiquetas generando los archivos que podrán ser cargados posteriormente.

Por otro lado, en la parte superior, hay unos datos que ayudan al usuario:

- Un checkbox para ocultar o mostrar los bordes de las regiones de la segmentación.
- El número del cuadro actual en el que se encuentra el usuario.
- El número de regiones en los que está segmentado el cuadro.

Además, en el lado derecho de la interfaz el usuario puede ver las etiquetas asignadas en ese cuadro y pulsando en cada una de ellas se seleccionará la región correspondiente a esa etiqueta.

4.1.2. Herramienta actual

En la nueva interfaz de la herramienta actual se observan cambios observables en la Figura 4.2 para poder realizar las actualizaciones realizadas sobre la herramienta base [1].

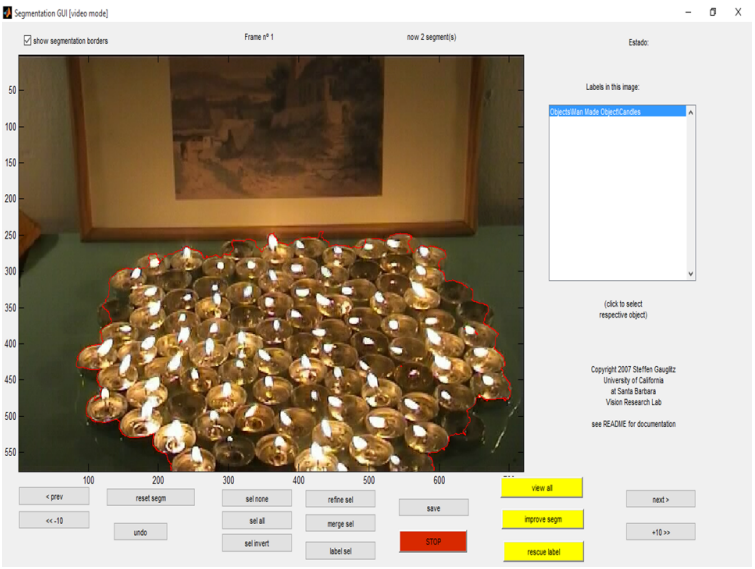


Figura 4.2: Interfaz de la herramienta desarrollada

En esta interfaz se observan fácilmente los botones nuevos ya que son los botones de colores. Estos tienen las siguientes funcionalidades:

- El botón *view all* abre un desplegable en el que aparecen todas las etiquetas asignadas durante todo el vídeo (independientemente de que aparezca en un sólo cuadro o en todo) para que el usuario seleccione una de ellas y se reproduzca el vídeo con la región asociada a esa etiqueta resaltada.

- El más importante de todos es el botón *improve segm* ya que es el encargado de, tras pulsar *save*, propagar esa información hacia los cuadros posteriores mientras va mostrando por pantalla los cambios realizados cuadro a cuadro.

- Hay casos en los que al modificar la segmentación con los botones $refine\ sel$ y *merge sel* desaparece la etiqueta. En estos casos, al pulsar el botón *rescue label* se recupera la etiqueta que está en el cuadro anterior pero no en el actual. Se la asigna a la región del cuadro actual que contenga el centro geodésico (Cg_j) de esa región j etiquetada en el cuadro anterior.

- El botón de *STOP* detiene la propagación por el usuario para que éste modifique manualmente el cuadro en el que se ha detenido. También detiene la

reproducción que realiza el botón *view all*.

Además de estos botones se ha añadido un campo de texto en la esquina superior derecha en la que se muestra el estado de la herramienta, como que se está propagando, que el usuario la ha parado o que se ha detectado un objeto entrante en la escena.

4.2. Descripción de la metodología de evaluación

Para evaluar la herramienta actual se han realizado una serie de ejercicios con ella y con la herramienta base [1] para comparar los resultados y ver como ha mejorado. Además estos ejercicios también se han pedido realizar a diez usuarios para que prueben la herramienta y posteriormente rellenen un cuestionario para ver cuáles son las impresiones de los usuarios y unas valoraciones de cero a cinco para que puntúen la herramienta en diferentes categorías.

4.3. Descripción de los experimentos cuantitativos realizados

A partir de trece vídeos se han etiquetado una serie de regiones con la herramienta base [1] y con la actual, midiendo el tiempo que se tarda en ambos casos y el número de cuadros en los que ha intervenido el usuario. Se considera que el usuario interviene en un cuadro ya sea sólo para pulsar el botón *improve segm* en un falso positivo de la deteción de objeto entrante o para modificar la segmentación del cuadro. Con el número de cuadros en los que se ha interaccionado tanto en la herramienta anterior como en la herramienta actual se obtiene el valor $S_{interaccion}$:

$$S_{interaccion} = 1 - (\frac{Interacciones_{actual}}{Interacciones_{inicial}}) \tag{4.1}$$

De esta forma se obtiene un valor más cercano a 1 cuando el usuario ha interactuado en el menor número de cuadros del vídeo e igual a 0 cuando ha interactuado en todos y por lo tanto es equivalente la herramienta anterior que la actual.

Por otro lado, una vez se hayan realizado los ejercicios tanto con la herramienta actual como con la herramienta base [1] con los segmentadores de Felzenszwalb [3] y de superpíxeles[2], se comprueba el ajuste de la región etiquetada en comparación de la herramienta actual con la base [1]. Para ello se usan las máscaras de las regiones etiquetadas, con la herramienta base [1] ($Minicial_i$) y con la actual ($Mactual_i$), de forma que el valor de ajuste σ_i se obtiene de esta forma:

$$\sigma_i = \frac{Mactual_i \cap Minicial_i}{Mactual_i \cup Minicial_i} \tag{4.2}$$

Con $i \in [1, N]$ siendo N el número de cuadros del vídeo y $N_q$el número de cuadros con anotaciones. Con este valor σ_i calculado a lo largo del vídeo se obtienen los siguientes valores para cada vídeo:

$$\overline{S_{ajuste}} = \frac{1}{N_q} \sum_{i=1}^{N_q} \sigma_i \tag{4.3}$$

$\overline{S_{ajuste}}$ representa la media de los valores de ajuste obtenidos en un vídeo, siendo siempre un valor entre 0 y 1, siendo el mejor resultado cuanto más cercano a 1 sea.

$$\hat{S_{ajuste}} = \max_i \{\sigma_i, i \in [1, N]\} \tag{4.4}$$

Este valor es el valor máximo alcanzado por el valor de ajuste, que dado que tanto en la herramienta base [1] como en la actual, el primer cuadro hay que procesarlo manualmente, esta labor será igual en ambos casos y por lo tanto el valor de ajuste valdrá 1, siendo el máximo alcanzable.

$$\check{S_{ajuste}} = \min_i \{\sigma_i, i \in [1, N]\} \tag{4.5}$$

Por otro lado se obtiene el valor mínimo del valor de ajuste que, al igual que la media, cuanto más cercano a 1, mejor resultado, ya que la anotación manual y la de la herramienta serán muy similares.

Mediante estos valores de ajuste se pueden representar una gráfica de barras de error para observar la calidad en cada vídeo.

Para hallar estos datos se realizan unos ejercicios con unos vídeos de 25 cuadros con distintas características. En las tablas D.1 y D.2 del Anexo D se muestran el primer y el último cuadro de los 25 cuadros de cada vídeo de prueba.[20][21].

4.4. Resultados de los experimentos cuantitativos

En la tabla 4.1 podemos ver los resultados obtenidos del $S_{interaccion}$ tras los ejercicios realizados con el segmentador en superpíxels[2] y el de Felzenszwalb[3] respectivamente.

	Superpíxels[2]			Felzenszwalb[3]		
	Cuadros H. Base [1]	Cuadros H. Actual	$S_{interaccion}$	Cuadros H. Base [1]	Cuadros H. Actual	$S_{interaccion}$
Atasco	25	3	0.88	25	3	0.88
Bandera	25	3	0.88	25	5	0.80
BusyBoulvard	25	5	0.80	25	7	0.72
Caldero	25	5	0.80	25	7	0.72
ContinuousPan	25	4	0.84	25	4	0.84
Fountain	25	3	0.88	25	3	0.88
Molino	25	2	0.92	25	5	0.80
Office	25	7	0.72	25	7	0.72
Playa	25	2	0.92	25	2	0.92
Rio	25	3	0.88	25	5	0.80
Traffic	25	10	0.60	25	10	0.60
Velas	25	4	0.84	25	7	0.72
WetSnow	25	5	0.80	25	5	0.80
MEDIA	**25**	**4.31**	**0.83**	**25**	**5.38**	**0.78**

Tabla 4.1: Tabla de cuadros en los que el usuario interacciona (Superpíxels[2] y Felzenszwalb[3])

Por otro lado, en la tabla 4.2 podemos ver los resultados obtenidos del S_{ajuste} también con ambos segmentadores.

	$\overline{S_{ajuste}}$	
	Superpíxels[2]	Felzenszwalb[3]
Atasco	0.9372	0.8431
Bandera	0.9579	0.8474
BusyBoulvard	0.8669	0.9024
Caldero	0.9011	0.6228
ContinuousPan	0.9226	0.9233
Fountain	0.9207	0.9548
Molino	0.9407	0.8918
Office	0.8255	0.9198
Playa	0.9024	0.9685
Rio	0.9609	0.8114
Traffic	0.7402	0.9222
Velas	0.9404	0.9718
WetSnow	0.9241	0.9611
MEDIA	**0.9033**	**0.8877**

Tabla 4.2: Tabla de ajustes (Superpíxels[2] y Felzenszwalb[3])

Finalmente [1] podemos realizar una gráfica como en la Figura 4.3 con los resultados obtenidos para ver cómo se acerca cada vídeo al resultado óptimo, es decir, al punto (1,1).

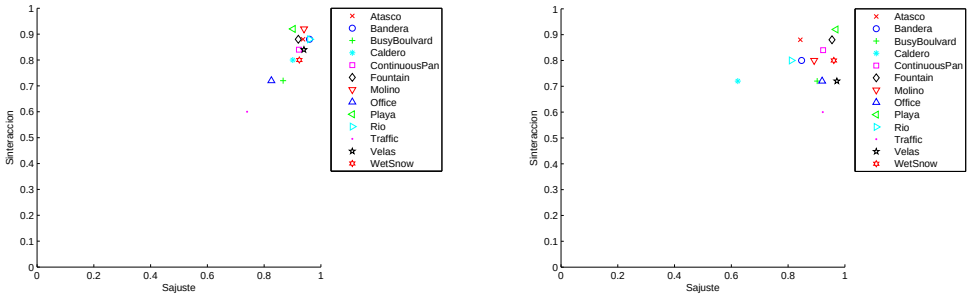


Figura 4.3: Relación $S_{interaccion}/\overline{S_{ajuste}}$ con el segmentador en superpíxels[2] a la izquierda y con Felzenszwalb[3] a la derecha

Estas gráficas representan la relación entre $S_{interaccion}$ y $\overline{S_{ajuste}}$ que como se ha comentado cuanto más cercano sea el resultado al punto (1,1) es más óptimo, mientras que el punto (0,0) sería el peor.

Además de esta relación podemos obtener unas gráficas representadas en la Figura 4.4 para ver cómo varía el S_{ajuste} con la media, el máximo y mínimo.

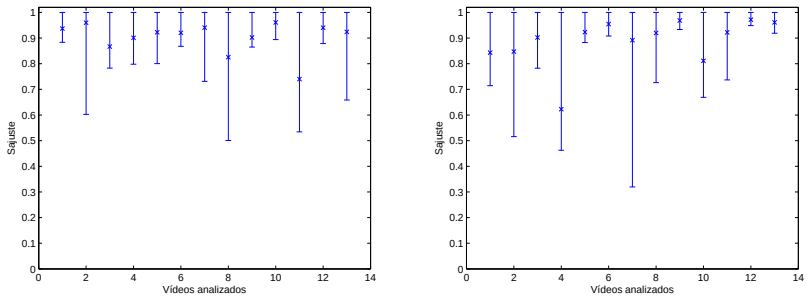


Figura 4.4: Gráfica de error con superpíxels[2] a la izquierda y con Felzenszwalb[3] a la derecha, donde en el eje X se observa cada uno de los vídeos analizados y en el eje Y el máximo y el mínimo del S_{ajuste} y la media marcada con una equis

4.5. Discusión de los resultados cuantitativos

Con estos resultados obtenidos se observa la mejora que se obtiene en la reducción de la intervención del usuario. Así podemos decir que la nueva herramienta facilita y mejora la labor del usuario.

Además se puede observar cómo se obtiene un resultado similar a realizar el etiquetado manualmente con la herramienta base [1].

Cabe destacar, que aunque con el segmentador de Felzenszwalb[3] se obtienen también buenos resultados, se obtienen unas regiones más fieles al objeto con el segmentador en superpíxels[2] implementado. Éste se ajusta mejor a los bordes de los objetos y de forma más regular, mientras que el de Felzenszwalb[3] genera regiones más irregulares que en ocasiones no se ajustan correctamente a los bordes. En la Figura 4.5 podemos ver un ejemplo:

Figura 4.5: Comparativa de segmentadores, a la izquierda Felzenszwalb [3] y a la derecha superpíxels [2]

Como se ve en la Figura 4.5, el segmentador implementado de superpíxels [2] se adapta mejor a los objetos.

Además, realizando los ejercicios podemos extraer también varias conclusiones acerca del rendimiento de la herramienta actual:

- La mayor ventaja se encuentra en el etiquetado de regiones que con la segmentación inicial está dividida en muchas regiones (como sucede en los vídeos de *Atasco* y $Velas$), lo que supone una gran labor por parte del usuario para unir todas las regiones deseadas. La herramienta actual permite hacer esta parte más dura en el primer cuadro y en posteriores simplemente realizar alguna pequeña modificación de menor peso que en el primer cuadro si es necesario.

- El vídeo de *Rio* se encuentran problemas al mover la cámara ya que hay determinadas regiones del agua o del campo, que al estar juntas, se unen y el usuario debe interactuar.

- En $Traffic$ nos encontramos ante un vídeo que dada su mala calidad no están bien definidas las regiones que separan el coche de la carretera, lo que provoca que durante la propagación la región que define al coche no lo siga con precisión lo que obliga a intervenir al usuario para solucionarlo.

A pesar de cualquier inconveniente que haya surgido siempre se ha tardado menos tiempo y se ha intervenido en menos cuadros, por lo que se ha conseguido una mejor experiencia con la herramienta.

4.6. Descripción de los experimentos con usuarios

La herramienta se ha enviado ha diez usuarios para que realicen los mismos ejercicios que en el apartado 4.3. pero sin cronometrarse ni contando los cuadros en los que realiza modificaciones ya que hay que tener en cuenta el tiempo que lleva a los usuarios entender y realizar un buen uso de la herramienta.

En su lugar se ha pedido a los usuarios que, tras las pruebas realizadas, rellenaran un cuestionario con unas valoraciones al final de distintos apartados de 0 a 5.

El cuestionario puede verse en el Anexo C.

4.7. Resultados de los experimentos con usuarios

En la tabla 4.3 se muestran los resultados recogidos de los usuarios.

	Resultados de experimentos por usuarios						
	Diseño	Fácil	M. Interfaz	M. Usabilidad	M. Eficiencia	M. Comodidad	Satisf. General
U1	5	3	4	4	5	5	5
U2	4	2	3	5	4	4	4
U3	5	4	4	4	4	5	5
U4	4	2	4	2	4	3	4
U5	4	3	4	4	5	4	4
U6	5	1	4	3	5	5	4
U7	4	3	4	5	4	4	4
U8	4	3	4	3	4	3	4
U9	5	4	3	4	5	5	5
U10	5	4	3	4	5	4	4
MEDIA	**4.5**	**2.9**	**3.7**	**3.8**	**4.5**	**4.2**	**4.3**

Tabla 4.3: Tabla valoraciones

4.8. Discusión sobre los experimentos con usuarios

Impresión general de la herramienta Viendo la tabla de valoraciones se observa que lo que menos han valorado los usuarios es la facilidad, ya que comentan que les

han surgido muchas dudas y al principio debían consultar las instrucciones constantemente. Por otro lado el diseño de la interfaz lo han considerado bueno pero luego se ha considerado que no ha sufrido una gran mejora respecto a la inicial. Respecto a la usabilidad también consideran que no se ha conseguido una gran mejora pues encuentran dificultades en la herramienta actual y en la inicial. Aún así consideran que la gran mejora de la herramienta actual es la comodidad, ya que ven reducido enormemente la cantidad de trabajo sin reducir la calidad y en mucho menos tiempo resultando la herramienta actual de una buena satisfacción general.

En términos generales, los usuarios han encontrado útil la herramienta, aunque al principio les ha costado comprender cómo funciona luego han sabido usarla correctamente tras la práctica. Además consideran buenas y correctas las mejoras ya que facilitan enormemente la labor del usuario.

Problemas encontrados Algunos usuarios han tenido dificultades a la hora de ejecutar la herramienta, ya que algunas funciones del segmentador de superpíxels[2] y del detector de objetos entrantes sólo funcionan en un sistema operativo Windows de 64 bits.

En las respuestas del cuestionario los usuarios reflejan que de primeras les ha costado entender la interfaz a pesar de las instrucciones y lo que más útil han visto de estas son el ejemplo de utilización, pero que quedaría más claro con imágenes.

Para que se le facilite la labor al usuario se pueden implementar ventanas emergentes cuando se realicen determinadas acciones. Por ejemplo, ha habido algún usuario que ha pulsado el botón *label sel* para poner una etiqueta sin tener seleccionada ninguna región, en este caso y similares se pueden implementar ventanas emergentes que avisen al usuario de su error.

Han sugerido también la implementación de un botón que elimine la etiqueta seleccionada ya que lo encuentran de mucha utilidad.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

La herramienta actual permite al usuario etiquetar un objeto de una escena y seguirlo a lo largo de un vídeo con mucho menos trabajo que en la herramienta base [1]. Mientras que con la inicial el usuario realizaba el mismo trabajo en cada cuadro, con la herramienta actual el usuario realiza ese trabajo más costoso en el primer cuadro, luego sólo extrapola esa información hacia los cuadros siguientes. Además el usuario puede interactuar en algunos cuadros parando el proceso o si la herramienta detecta un nuevo objeto en la escena y realizar los cambios convenientes. Estos cambios son mucho menores que en el primer cuadro, ya que son de pequeños detalles. Todo esto provoca que la labor sea mucho más rápida (ver Anexo A) y como se ha visto en los experimentos realizados, casi con la misma eficiencia que si el usuario hubiera realizado los cambios en cada cuadro.

Con todo esto podemos llegar a la conclusión de que se ha conseguido una herramienta muy útil a la hora de etiquetar objetos de una escena de una forma cómoda y sencilla para el usuario.

5.2. Trabajo futuro

La herramienta puede avanzar con muchas mejoras de muchos tipos:

- Como se ha dicho añadir un botón para eliminar las etiquetas asignadas.
- Añadir la opción al usuario de realizar la propagación con detector de objetos entrantes o sin él, para que la propagación sea más rápida.

- Usar otros tipos de segmentadores si dan mejores resultados, como un segmentador espacio-temporal que realiza una segmentación similar en cada cuadro.
- Añadir a la interfaz la posibilidad de que el usuario pueda introducir el número del cuadro al que quiere ir, sin usar los botones de *prev* y *next*.
- Usar modelos de objetos para que al asignar etiquetas aparezcan las que son más probables que correspondan con esa forma de la región.
- Añadir a la reproducción del vídeo la opción de reproducirlo en imágenes de colores donde cada color se corresponda con cada región.

Bibliografía

[1] S. Gauglitz, "Semi-automatic image segmentation & annotation tool," 2007. xiii, xiii, xiii, 2, 6, 7, 8, 9, 11, 12, 13, 14, 15, 18, 22, 25, 29, 31, 32, 33, 34, 35, 36, 39

[2] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," March 2016. xiii, xiii, xiii, xiii, xv, xv, 15, 16, 25, 32, 33, 34, 35, 36, 38, 47

[3] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, September 2004. xiii, xiii, xv, xv, 15, 16, 32, 33, 34, 35, 36, 47

[4] S. Dasiopoulou, E. Giannakidou, G. Litos, P. Malasioti, and Y. Kompatsiaris, "A survey of semantic image and video annotation tools," 2011. 1, 9

[5] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, pp. 1–21, 2011. 10.1007/s11263-012-0564-1. 6, 7, 8

[6] M. A. Serrano, J. García, M. A. Patricio, and J. M. Molina, *Interactive Video Annotation Tool*. 2010. 7, 8

[7] P. Schallauer, S. Ober, and H. Neuschmied, "Efficient semantic video annotation by object and shot re-detection," 2008. 7, 8

[8] B. Manjunath, P. Salembier, and T. Sikora, *Introduction to MPEG 7: Multimedia Content Description Language*. 2002. 9

[9] G. Gualdi, "Contextual information and covariance descriptors for people surveillance," 2011. 9

[10] S. Oh, A. Hoogs, M. Turek, and R. Collins, "Content-based retrieval of functional objects in video using scene context," 2010. 9

[11] P. Kumar, "Framework for real-time behavior interpretation from traffic video," 2005. 9

[12] R. Mazzon and A. Cavallaro, "Multi-camera tracking using a multi-goal social force model," 2013. 9

[13] F. Jianga, J. Yuanc, S. A. Tsaftaris, and A. K. Katsaggelos, "Anomalous video event detection using spatiotemporal context," 2011. 9

[14] J. K. Santiago Manen and, M. Guillaumin, and L. V. Gool, "Appearances can be deceiving: Learning visual tracking from few trajectory annotations," 2014. 10

[15] T. Bouwmans, "Traditional and recent approaches in background modelling for foreground detection: An overview," 2014. 10

[16] A. Shahrokni, T. Drummond, and P. Fua, "Texture boundary detection for real-time tracking," 2004. 10

[17] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer, "A survey of urban reconstruction," 2013. 10

[18] Y. Witmaar, "Desarrollo de herramienta para la anotación manual de secuencias de vídeo," 2015. 12

[19] C. Liu, "Beyond pixels: Exploring new representations and applications for motion analysis," 2009. 23

[20] R. Péteri, S. Fazekas, and M. J. Huiskes, "DynTex : a Comprehensive Database of Dynamic Textures," vol. doi: 10.1016/j.patrec.2010.05.009, 2010. http://projects.cwi.nl/dyntex/. 33

[21] P.-M. Jodoin, P. Ishwar, J. Konrad, and F. Porikli, "A video database for testing change detections algorithms," 2014. 33

Apéndice A

Medidas de tiempos

Cuando se han realizado los ejercicios de etiquetado en los 13 vídeos se ha medido el tiempo que se ha invertido para ver como un usuario que conoce la herramienta invierte menos tiempo con la herramienta actual.

	Resultados de experimentos		
	Tiempo Herramienta Anterior	Tiempo Herramienta Actual	Mejora
Atasco	44:07	07:13	83.64 %
Bandera	38:20	08:30	77.82 %
BusyBoulvard	34:35	06:19	81.73 %
Caldero	25:23	07:38	69.92 %
ContinuousPan	27:50	11:50	57.48 %
Fountain	36:15	04:28	87.68 %
Molino	21:40	06:01	72.23 %
Office	16:00	06:46	57.71 %
Playa	30:35	07:01	77.06 %
Río	45:50	25:08	45.16 %
Traffic	19:35	12:42	35.15 %
Velas	1:05:20	06:23	90.23 %
WetSnow	40:50	27:55	31.63 %
MEDIA	**34:20**	**10:36**	**69.12 %**

Tabla A.1: Tabla de tiempos

Como se puede observar, se obtiene una mejora media del 69.12 % en el tiempo invertido en los ejercicios. Por consiguiente, se puede decir que la herramienta permite mucha más rapidez en la labor del usuario lo que le facilita la labor.

Apéndice B

Manual del programador

Para ejecutar la herramienta se necesitan varias cosas. Lógicamente se necesita el vídeo que se puede añadir a la carpeta *videos*. Además se necesitan los frames del vídeo para la segmentación que se deben guardar en una carpeta (para facilitar las cosas se nombra la carpeta como al vídeo) y a su vez se crea una carpeta con el nombre $files$ dentro de ésta, donde se van a almacenar los ficheros que se guarden. Estos ficheros son la imagen de segmentación (.gif) y el fichero de etiquetas (.mat) donde se almacena el nombre de la etiqueta, el valor de sus píxeles, el centro geodésico y su tamaño. Además de estos ficheros, en la carpeta superior se almacenan los ficheros de la segmentación inicial de superpíxels [2] (.mat) o de Felzenszwalb [3] (.ppm). Todos estos ficheros se pueden borrar ejecutando la función $delete\_files(nombre\_carpeta)$ para los archivos guardados y $delete\_segm(nombre\_carpeta)$ para los ficheros de segmentación inicial.

Para iniciar la herramienta vale con ejecutar el fichero *demo.m* escribiendo el nombre de la carpeta (que coincide con el del vídeo) y se ejecuta el comando siguiente:

$segmentation\_ui(loaddir('videos', [folder,' .avi']), ...$

$...loadstrings('categories.txt'), 1,' video', folder);$

El fichero *categories.txt* es el documento de texto donde se encuentran las etiquetas disponibles para asignar a los objetos del vídeo.

El parámetro $folder$ es el nombre de la carpeta que en este caso coincide con el del vídeo.

El número indica el número del frame que se quiere abrir.

El parámetro $'video'$ indica la modalidad que para este caso sólo nos interesa el modo de vídeo.

Tras ejecutar esta función se abrirá la interfaz de la herramienta actual de la que ya se ha hablado.

Apéndice C

Cuestionario a los usuarios

Este es el cuestionario que se les facilitó a los usuarios:
INTERFAZ

1. ¿Las instrucciones le han parecido adecuadas? ¿Echa algo en falta?
2. ¿Le parece adecuada la cantidad de botones que se observan en pantalla? ¿Echa en falta algún botón que realice otras acciones? ¿Cuál?
3. ¿A primera vista ha sabido distinguir cuáles eran los botones más relevantes? ¿Por qué?

USABILIDAD

1. ¿Existen elementos que le permitan saber exáctamente qué proceso está realizando?
2. ¿Sabe volver atrás tras una acción no deseada?
3. ¿Le parece difícil abrir la herramienta?

COMPARATIVA

1. ¿Se distinguen fácilmente las actualizaciones realizadas? ¿Por qué?
2. Tras una primera mirada, ¿le queda claro cuál es el objetivo de la aplicación? ¿Qué ventajas ofrece? ¿Las puede enumerar?
3. ¿Cómo calificaría las mejoras: demasiadas, justas o insuficientes? (De ser demasiadas especificar qué eliminaría y en el caso de insuficientes qué añadiría)

IMPRESIÓN GENERAL/GLOBAL

1. ¿Cree que esta aplicación puede ser de utilidad?
2. ¿Qué es lo que más le llamó la atención positívamente sobre su utilidad? ¿Y negativamente?
3. ¿Qué es lo que más dificultad le ha generado?

COMPARATIVA CUANTITATIVA

1. De 0 a 5, ¿cómo valoraría el diseño de la interfaz?
2. De 0 a 5, ¿cómo valoraría la facilidad de uso?
3. De 0 a 5, ¿cuánto cree que ha mejorado la aplicación?
    - Interfaz
    - Usabilidad
    - Eficiencia
    - Comodidad
4. De 0 a 5, ¿cómo valoraría su satisfacción general con la aplicación?

Apéndice D

Ejercicios de anotación

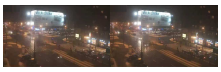
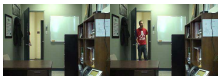
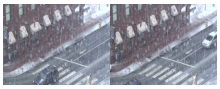
Dataset: Objetos no dinámicos		
Atasco: Etiquetar carretera 	BusyBoulvard: Etiquetar carretera 	ContinuousPan: Etiquetar carretera 
Office: Etiquetar puerta 	Traffic: Etiquetar carretera y coche 	WetSnow: Etiquetar carretera 

Tabla D.1: Dataset objetos no dinámicos

En estos vídeos se etiquetan objetos estáticos del entorno (sin tener en cuenta el movimiento de la cámara).

En el vídeo *Atasco* se observa un entorno muy común de una autopista con tráfico denso, lo que dificulta el etiquetado de la carretera por su irregularidad. Además la cámara gira hacía la derecha lo que provoca que aparezcan nuevos coches que hay que tener en cuenta para el etiquetado de la carretera.

En *BusyBoulvard* nos encontramos un entorno nocturno, lo que provoca que los objetos tengan un color y un tono similar, además de encontrar mucho movimiento en la escena.

En el vídeo *continuousPan* volvemos a encontrar una carretera, esta vez no transitada, en la que nuevamente la cámara gira hacia la izquierda.

El vídeo $Office$ produce una dificultad, ya que por la puerta que hay que etiquetar entra una persona, lo que provoca que quizás alguna región de la persona se una con la de la puerta.

El vídeo *Traffic* tiene mucha dificultad, ya que el vídeo es de baja calidad y con jitter, lo que provoca que los bordes de los objetos no estén bien definidos y esto hace que las regiones no tengan gran precisión, por lo que el coche que atraviesa la carretera durante el vídeo no puede ser separado de la carretera por sus bordes exactamente.

En *WetSnow* aparece una escena en la que nieva, lo que dificulta la segmentación ya que la nieve varía la escena en cada cuadro y empeora la calidad de la imagen.

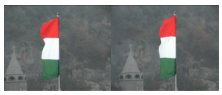
Dataset: Objetos dinámicos		
Bandera: Etiquetar bandera 	Caldero: Etiquetar humo 	Fountain: Etiquetar agua 
Molino: Etiquetar bandera 	Playa: Etiquetar mar 	Rio: Etiquetar agua 
	Velas: Etiquetar velas como conjunto 	

Tabla D.2: Dataset objetos dinámicos

Estos vídeos contienen objetos dinámicos en la escena por lo que se van a usar para comprobar cómo responde la herramienta.

El vídeo *Bandera* muestra dificultad por el continuo movimiento de la tela y por el parecido de la franja verde con el fondo lo que puede dar problemas al segmentador que los considere una misma región.

En *Caldero* surge la dificultad de etiquetar el humo que no se puede considerar un objeto bien definido por sus bordes y que se puede confundir con el fondo.

En el vídeo de *Fountain* sucede algo parecido al anterior, ya que el agua que expulsa la fuente no tiene unos bordes definidos y bastante variables en cada cuadro.

En el vídeo *Molino* nos encontramos nuevamente con una bandera, la diferencia es que aparece un aspa de molino que oculta la bandera, lo que se puede comprobar si la herramienta, por si sola, elimina la etiqueta de la bandera cuando se oculta.

En la escena del vídeo *Playa* el mar que hay que etiquetar está en movimiento lo que puede hacer que varíe la segmentación y además aparecen muchos elementos en la escena que pueden dificultar el etiquetado, como unas personas que aparecen por

detrás del vehículo y ocultan una zona etiquetada como mar.

El vídeo *Rio* tiene la dificultad de no tener bien definidos los bordes de las zonas verdes con el agua, además de que la imagen de la escena se va acercando y moviéndose, lo que provoca que se pierda calidad en la imagen.

En el vídeo *Velas* aparecen un conjunto de velas, que al ser etiquetadas en conjunto aparece la dificultad de hacerlo cuadro a cuadro por existir muchas regiones internas al conjunto, pero esto no sucederá con la herramienta actual, ya que esta labor más pesada sólo se realiza en el primer cuadro.

En las siguientes figuras se muestra las regiones que se anotan marcadas en rojo:

Figura D.1: Ejercicio de etiquetado de *Atasco*

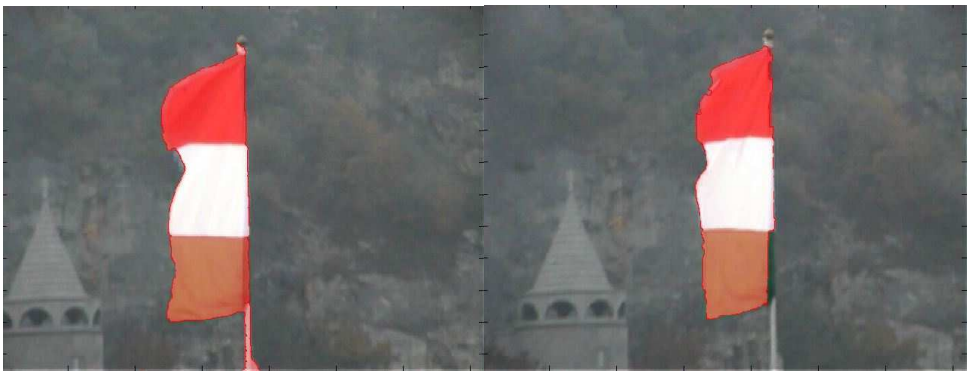


Figura D.2: Ejercicio de etiquetado de *Bandera*

Figura D.3: Ejercicio de etiquetado de *BusyBoulvard*

Figura D.4: Ejercicio de etiquetado de *Caldero*

Figura D.5: Ejercicio de etiquetado de *ContinuousPan*

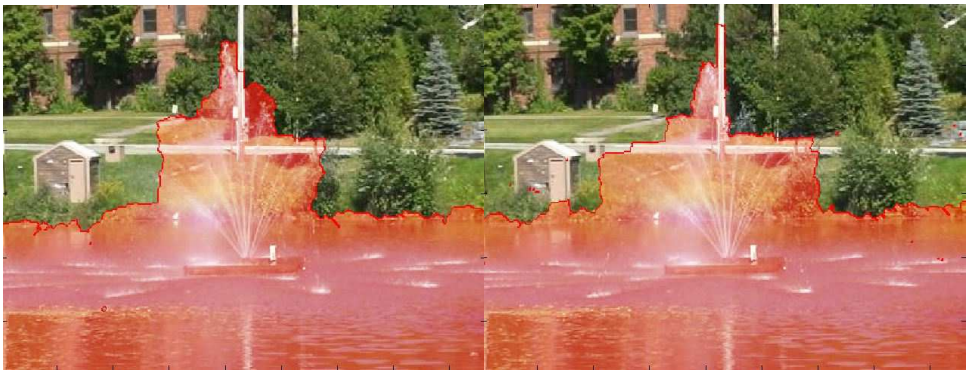


Figura D.6: Ejercicio de etiquetado de *Fountain*

Figura D.7: Ejercicio de etiquetado de *Molino*

Figura D.8: Ejercicio de etiquetado de *Office*

Figura D.9: Ejercicio de etiquetado de *Playa*

Figura D.10: Ejercicio de etiquetado de *Rio*

Figura D.11: Ejercicio de etiquetado de $Traffic$

Figura D.12: Ejercicio de etiquetado de *Velas*

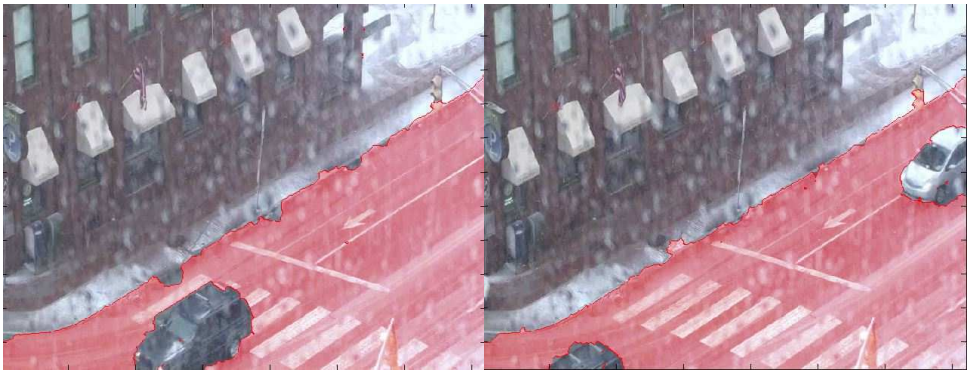


Figura D.13: Ejercicio de etiquetado de *WetSnow*

Apéndice E

Código desarrollado

Esta es la función inicial a la que se llama para ejecutar la herramienta que recibe todos los parámetros.

```
function varargout = segmentation_ui(varargin)
    gui_Singleton = 1;
    gui_State = struct('gui_Name',       mfilename, ...
                       'gui_Singleton',  gui_Singleton, ...
                       'gui_OpeningFcn', @segmentation_ui_OpeningFcn, ...
                       'gui_OutputFcn',  @segmentation_ui_OutputFcn, ...
                       'gui_LayoutFcn',  [] , ...
                       'gui_Callback',   []);
    if nargin && ischar(varargin{1})
        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargout
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end

function segmentation_ui_OpeningFcn(hObject, eventdata, handles, varargin)

    handles.images     = varargin{1}; % first  parameter: images
    handles.categories = varargin{2}; % second parameter: categories
    if length(varargin)==2
        i = 1;
        handles.mode = 'image';
        handles.labels = struct([]);
```

```
27          handles.segmentation = struct([]);
28          handles.polygons = struct([]);
29          folder='NULL';
30      elseif length(varargin)==5
31          i = varargin{3};
32          handles.mode = varargin{4};
33          handles.labels = struct([]);
34          handles.segmentation = struct([]);
35          handles.polygons = struct([]);
36          folder = varargin{5};
37          if strcmp(handles.mode, 'video')==true
38              file = handles.images{1};
39              handles.video = mmread(file);
40          end
41      elseif length(varargin)==6
42          i = varargin{3};
43          handles.mode = varargin{4};
44          handles.labels = struct([]);
45          handles.segmentation = struct([]);
46          handles.polygons = struct([]);
47          folder = varargin{5};
48          handles.video = varargin{6};
49      elseif length(varargin)==8
50          i = varargin{3};
51          handles.mode = varargin{4};
52          handles.labels = varargin{6};
53          handles.segmentation = varargin{7};
54          handles.polygons = varargin{8};
55          folder = varargin{5};
56          if strcmp(handles.mode, 'video')==true
57              file = handles.images{1};
58              handles.video = mmread(file);
59          end
60      elseif length(varargin)==9
61          i = varargin{3};
62          handles.mode = varargin{4};
63          handles.labels = varargin{6};
64          handles.segmentation = varargin{7};
65          handles.polygons = varargin{8};
66          folder = varargin{5};
67          handles.video = varargin{9};
68      else
69          close(handles.figure1);
70          return
71      end
72      handles.colormap = randomcolormap();
```

```
73      handles.oldsegm = 0;
74      handles.output = hObject; % Choose default command line output for
            segmentation_ui
75      handles.edit=false;
76      handles.folder=folder;
77      guidata(hObject, handles); % Update handles structure
78
79      initialize_image(hObject,i,folder);
```

Esta función a su vez llama a la función que inicia la imagen que lee los datos que hay guardados o los crea si no los hay

```
1   function initialize_image(handle,i,folder)
2
3       d=guidata(handle);
4       d.i=i;
5       addpath(folder);
6
7       if strcmp(d.mode, 'video')==false && strcmp(d.mode, 'image')==false
8          close(d.figure1);
9          return
10      end
11
12      if strcmp(d.mode, 'image')==true
13          file = d.images{i};
14          precompute_segmentations(file);
15          d.image        = imread(file);
16
17          d.length = length(d.images);
18
19          [ly,lx,depth]=size(d.image);
20
21          d.oldsegm      = 0;
22          d.segmentation= load_segmentation(file, d.mode, i, d.folder);
23          d.oldrefine    = 0;
24
25          if exist([file '.labels.mat'],'file')
26            labels=load([file '.labels.mat']);
27            if length(d.labels)==0
28                for i=1:length(labels.labels)
29                    d.labels{1,i}.label = labels.labels{1,i}.label;
30                    d.labels{1,i}.segmenttoken = labels.labels{1,i}.
                          segmenttoken;
31                end
32            end
```

```
33          end
34
35          if exist([file '.polygons.mat'],'file')
36            polygons=load([file '.polygons.mat']);
37            if length(d.polygons)==0
38                d.polygons = polygons.polygons;
39            end
40          end
41      end
42
43      if strcmp(d.mode, 'video')==true
44          file = d.images{1};
45          d.image = d.video.frames(i).cdata;
46          precompute_segmentations_video(folder);%segmenta los frames del
                video
47          d.length = length(d.images);
48          [ly,lx,depth]=size(d.image);
49
50          d.oldsegm = 0;
51          d.file=file;
52          d.segmentation = load_segmentation(file, d.mode, i, folder);%se
                carga la segmentación del frame i
53          d.oldrefine = 0;
54
55
56          l=importdata('list.txt');
57          img=l{i};
58                  %se cargan todas las etiquetas del frame i si existen
59          if exist([d.folder '\' 'files' '\' img '.labels.mat'],'file')
60            labels=load([d.folder '\' 'files' '\' img '.labels.mat']);
61            if length(d.labels)==0
62                for i=1:length(labels.labels)
63                    d.labels{1,i}.label = labels.labels{1,i}.label;
64                    d.labels{1,i}.segmenttoken = labels.labels{1,i}.
                          segmenttoken;
65                end
66            end
67          end
68
69          d.length = d.video.nrFramesTotal-1;
70          [ly,lx,depth] = size(d.image);
71          d.oldsegm = d.segmentation;
72          d.oldrefine = 0;
73      end
74
75      l=cell(1);
```

```
76      for i=1:length(d.labels), l{i}=d.labels{i}.label; end
77      set(d.listlabels,'String',l);
78      set(d.listlabels,'Value',1);
79
80      x=size(d.image,2);
81      y=size(d.image,1);
82      % si una región está seleccionada al iniciar el prev/next frame
            sigue seleccionada
83      if isfield(d,'selstatus')==1
84          if max(max(d.selstatus))==1
85              [centro_x,centro_y,~]=hallar_centro(d.selstatus,1,Inf);
86              token=d.segmentation(centro_x,centro_y);
87              d.selstatus   = zeros(y,x,'uint8');
88              d.selstatus(d.segmentation==token)=1;
89          else
90              d.selstatus   = zeros(y,x,'uint8');
91          end
92      else
93          d.selstatus   = zeros(y,x,'uint8');
94      end
95      d.refinelevel = ones (y,x,'uint8');
96      d.red         = ones (y,x,'uint8');
97
98      guidata(handle,d);
99      showimage(d);
```

Se llama a la función que realiza la segmentación inicial de los frames del vídeo que si ya se ha hecho vuelve. En este caso se realiza la segmentación por superpíxeles que se ha implementado y como se decía se realiza la eliminación de los bordes de las regiones.

```
1   function precompute_segmentations_video(folder)
2           % lee las imágenes de los frames si son png o jpg
3       if numel(dir([folder '\*.png']))>0
4           system(['dir .\' folder '\*.png /b  > list.txt']);
5       else
6           system(['dir .\' folder '\*.jpg /b  > list.txt']);
7       end
8
9       l=importdata('list.txt');
10      k=length(l);
11
12      addpath models %carpeta de segmentador superpíxels
13
14      model=load('models/forest/modelBsds'); model=model.model;
15      model.opts.nms=-1; model.opts.nThreads=4;
```

```
16      model.opts.multiscale=0; model.opts.sharpen=2;
17
18      for z=1:k
19
20          filename=[folder '/' l{z}];
21          if exist([filename '.segpre1.mat'],'file') && ...
22             exist([filename '.segpre2.mat'],'file') && ...
23             exist([filename '.segpre3.mat'],'file') && ...
24             exist([filename '.segpre4.mat'],'file') && ...
25             exist([filename '.segpre5.mat'],'file'), return;% si existen
                   se vuelve
26          else
27             display(['Segmentando frame n°' num2str(z)]);
28          end
29
30          for nivel=1:5
31              opts = spDetect;
32              opts.nThreads = 4;  %number of computation threads
33              opts.k = 8192/(2^nivel);        %controls scale of
                    superpixels (big k -> big sp)
34              opts.alpha = .5;    %relative importance of regularity
                    versus data terms
35              opts.beta = .9;     %relative importance of edge versus
                    color terms
36              opts.merge = eps;      %set to small value to merge nearby
                    superpixels at end
37
38              I = imread(l{z});
39              [E,~,~,segs]=edgesDetect(I,model);
40              [S,V] = spDetect(I,E,opts);%se obtiene la imagen de
                    regiones con bordes
41              cero=find(S==0);
42              while (cero>0)
43                  for i=1:size(S,1)
44                      for j=1:size(S,2)
45                          if(S(i,j)==0)
46                              if i<(size(S,1)/2)
47                                  if j<(size(S,2)/2)
48                                      if S(i+1,j+1)==0
49                                          S(i,j)=S(i+2,j+1);
50                                      else
51                                          S(i,j)=S(i+1,j+1);
52                                      end
53                                  else
54                                      if S(i+1,j-1)==0
55                                          S(i,j)=S(i+2,j-1);
```

```
56                                      else
57                                          S(i,j)=S(i+1,j-1);
58                                      end
59                                  end
60                              else
61                                  if j<(size(S,2)/2)
62                                      if S(i-1,j+1)==0
63                                          S(i,j)=S(i-2,j+1);
64                                      else
65                                          S(i,j)=S(i-1,j+1);
66                                      end
67                                  else
68                                      if S(i-1,j-1)==0
69                                          S(i,j)=S(i-2,j-1);
70                                      else
71                                          S(i,j)=S(i-1,j-1);
72                                      end
73                                  end
74                              end
75                          end
76                      end
77                  end
78                  cero=find(S==0);
79              end
80              segm=S;%se han quitado los bordes
81              save([folder '/' l{z} '.segpre' num2str(nivel) '.mat'],'segm
                    ');
82          end
83      end
```

Una vez creadas las imágenes de segmentación se cargan desde esta función.

```
1   function segm=load_presegmentation(imgname,level,mode,i)
2
3       if strcmp(mode, 'video')==true
4           l=importdata('list.txt');
5           imgname=l{i};
6       end
7       file=[imgname '.segpre' num2str(level) '.mat'];
8
9       loop=0;
10      while exist(file,'file')==0 %wait for file to be rendered
11        pause(1); %wait one second
12        loop=loop+1;
13        if loop>30
```

```
14          warning('segmentation_ui:waiting','waiting for 30 seconds now.
                Are you sure the image is being rendered?!');
15          loop=0;
16        end
17      end
18
19      seg=load(file);%carga la segmentación inicial
20      segm=seg.segm;
```

La función anterior es llamada por esta otra que carga la segmentación si no existe una generada por el usuario y debe llamarla para cargar la segmentación inicial.

```
1   function segm=load_segmentation(imgname, mode, i, folder)
2
3       file=[imgname '.seg.gif'];
4       if strcmp(mode, 'video')==true
5           l=importdata('list.txt');
6           imgname=l{i};
7       end
8       file2=[folder '\' 'files' '\' imgname '.seg.gif'];%la ruta es
            distinta en modo vídeo
9       if exist(file,'file')~=0
10          segm=uint32(imread(file));
11      elseif exist(file2,'file')~=0
12          segm=uint32(imread(file2));%sólo en vídeo
13      else
14          segm=load_presegmentation(imgname,1,mode,i);%si no existe
15      end
```

Para guardar el estado de la segmentación y las etiquetas se usa la función siguiente.

```
1   function save_Callback(hObject, eventdata, d)
2
3       [map,s] = convert2_8bit(d.segmentation);
4       d.segmentation=s;
5       map=unique(s);
6       nframe=d.i;
7
8       if strcmp(d.mode, 'image')==true
9           filename = d.images{d.i};
10          imwrite( s, [filename '.seg.gif']);
11      elseif strcmp(d.mode, 'video')==true
12          l=importdata('list.txt');
13          filename=l{d.i};
14          [ly,lx,depth]=size(s);
```

```
15          segmentations = zeros(ly, lx, 1, 1);
16          segmentations = im2uint8(segmentations);
17          imwrite( segmentations, [d.folder '\' 'files' '\' filename '.seg
                .gif']);%se guarda la segmentación
18      end
19
20      j=1;
21      savelabels=0;
22      for cont=1:length(d.labels)
23        mylabel=d.labels{cont}.segmenttoken;
24        if( numel(find(d.segmentation==mylabel,1)) )>0
25          savelabels=1;
26          labels(j)=d.labels(cont);
27          labels{j}.segmenttoken=mylabel;
28          tok(j)=labels{j}.segmenttoken;%valor de los píxeles de la
                región
29          [d.centro_x(j),d.centro_y(j),d.c(j)]=hallar_centro(segmentations
                ,tok(j),Inf);
30          labels{j}.centro_x=d.centro_x(j);%punto x del centro geodésico
31          labels{j}.centro_y=d.centro_y(j);%punto y del centro geodésico
32          labels{j}.tam=d.c(j);%tamaño de la región
33          guidata(hObject,d);
34          j=j+1;
35        else
36            if (length(d.labels)==1)
37                delete([d.folder '\files\' l{d.i} '.labels.mat']);
38            end
39        end
40      end
41
42      if savelabels ~= 1
43      %   warning('segmentation_ui:save_Callback','No labels saved!');
44      else
45        if strcmp(d.mode, 'image')==true
46            save([filename '.labels.mat'],'labels');
47        elseif strcmp(d.mode, 'video')==true
48            save([d.folder '\' 'files' '\' filename '.labels.mat'],'labels
                  ');%se guardan las etiquetas
49        end
50      end
51          %aquí se va a generar un fichero etiquetas.mat donde se
                almacenan cada etiqueta del vídeo
52          %y los valores que tienen las regiones en las que se encuentra
                la etiqueta o NaN si no se encuentra
53      if exist('labels','var')==1
54          if exist([d.folder '\files\etiquetas.mat'],'file')==0
```

```
55              for i=1:length(labels)
56                  matriz{i}.etiqueta=labels{i}.label;
57                  matriz{i}.valores=NaN(1,length(l));
58                  matriz{i}.valores(nframe)=labels{i}.segmenttoken;
59              end
60          else
61              l_labels=length(labels);
62              var=load([d.folder '\files\etiquetas.mat']);
63              matriz=var.matriz;
64              l_matriz=length(matriz);
65              for i=1:l_labels
66                  contiene=0;
67                  for j=1:l_matriz
68                      if strcmp(matriz{j}.etiqueta,labels{i}.label)==1
69                          matriz{j}.valores(nframe)=labels{i}.segmenttoken
                                ;
70                          contiene=1;
71                      end
72                  end
73                  if contiene==0
74                      l_matriz2=length(matriz);
75                      matriz{l_matriz2+1}.etiqueta=labels{i}.label;
76                      matriz{l_matriz2+1}.valores=NaN(1,length(l));
77                      matriz{l_matriz2+1}.valores(nframe)=labels{i}.
                            segmenttoken;
78                  end
79              end
80          end
81          save([d.folder '\files\etiquetas.mat'],'matriz');
82      end
83
84      d.labels = struct([]);
85      d.polygons = struct([]);
86      d.segmentation = struct([]);
87      guidata(hObject,d);
88      initialize_image(gcbo,d.i,d.folder);
```

La función que asigna etiquetas ha sido modificada para que no se repitan las etiquetas y así guardarlas por separado en el fichero *etiquetas.mat*.

```
1   function label_Callback(hObject, eventdata, d)
2
3       [map,d.segmentation] = convert2_8bit(d.segmentation);
4       segments=deleteduplicates(d.segmentation(d.selstatus == 1));
5       if numel(segments)<1, return, end
```

```
6
7       [sel,ok]=listdlg('ListString',d.categories,'SelectionMode','single',
            ...
8           'ListSize',[250 500],'Name','select label');
9       if ok ~= 1, return, end % user clicked 'cancel'
10
11      answer=getHierarchicalLabel(d.categories,sel);
12
13          % bucle para poner un número después de una etiqueta repetida
14      for i=1:length(d.labels)
15          if strcmp(answer,d.labels{i}.label)==1
16              ultimo_caracter=str2num(answer(end));
17              if isempty(ultimo_caracter)==1
18                  answer=[answer '1'];
19              else
20                  answer=[answer(1:end-1) int2str(ultimo_caracter+1)];
21              end
22          end
23      end
24
25      for s=1:length(segments) % for all selected segments
26
27        i=1;
28        while i<=length(d.labels) % already an entry existing?
29          if d.labels{i}.segmenttoken==segments(s), break, end
30          i=i+1;
31        end % if loop runs through, i is automatically pointing to next
              entry
32
33        d.labels{i}.segmenttoken = segments(s);
34        d.labels{i}.label = answer;
35      end
36
37      for i=1:length(d.labels), l{i}=d.labels{i}.label; end
38      set(d.listlabels,'String',l);
39      guidata(hObject,d);
```

Para propagar las etiquetas se usa la siguiente función que guarda la etiqueta de la región homóloga del frame anterior.

```
1   function extrapolar(hObject, eventdata, d)
2
3       l=importdata('list.txt');
4       orig=l{d.i};
5       lab=load([d.folder '\' 'files' '\' orig '.labels.mat']);
```

```
6
7       d.i=d.i+1;
8       d.segmentation=load_segmentation(d.file, d.mode, d.i, d.folder);
9       nframe=d.i;
10
11      filename=l{d.i};
12      segmentations=imread([d.folder '/files/' filename '.seg.gif']);
13      cont=1;
14      for j=1:length(lab.labels)
15          centro_x=lab.labels{j}.centro_x;
16          centro_y=lab.labels{j}.centro_y;
17          tam=lab.labels{j}.tam;
18          if centro_x==0 && centro_y==0
19              j=j+1;
20          else
21              tok(j)=segmentations(centro_x,centro_y);
22              [centro_x,centro_y,tam]=hallar_centro(segmentations,tok(j),
                    tam);
23              if centro_x==0 && centro_y==0% significa que el objeto salió
                     de la escena
24                  lab.labels{j}.segmenttoken=10000;
25                  lab.labels{j}.label='';
26                  j=j+1;
27              else
28                  lab_aux.labels{cont}.segmenttoken=tok(j);
29                  lab_aux.labels{cont}.label=lab.labels{j}.label;
30                  lab_aux.labels{cont}.centro_x=centro_x;
31                  lab_aux.labels{cont}.centro_y=centro_y;
32                  lab_aux.labels{cont}.tam=tam;
33                  cont=cont+1;
34              end
35          end
36      end
37      imwrite( segmentations, [d.folder '\' 'files' '\' filename '.seg.gif
            ']);
38      if cont>1% si se genera alguna etiqueta se guarda
39          labels=lab_aux.labels;
40          clear lab_aux;
41          save([d.folder '\' 'files' '\' filename '.labels.mat'],'labels')
                ;
42      end
43          % se actualiza etiquetas.mat
44      if exist([d.folder '\files\etiquetas.mat'],'file')==0
45          for i=1:length(labels)
46              matriz{i}.etiqueta=labels{i}.label;
47              matriz{i}.valores=NaN(1,length(l));
```

```
48              matriz{i}.valores(nframe)=labels{i}.segmenttoken;
49          end
50      else
51          if cont>1
52              l_labels=length(labels);
53              var=load([d.folder '\files\etiquetas.mat']);
54              matriz=var.matriz;
55              l_matriz=length(matriz);
56              for i=1:l_labels
57                  contiene=0;
58                  for j=1:l_matriz
59                      if strcmp(matriz{j}.etiqueta,labels{i}.label)==1
60                          matriz{j}.valores(nframe)=labels{i}.segmenttoken
                                ;
61                          contiene=1;
62                      end
63                  end
64                  if contiene==0
65                      l_matriz2=length(matriz);
66                      matriz{l_matriz2+1}.etiqueta=labels{i}.label;
67                      matriz{l_matriz2+1}.valores=NaN(1,length(l));
68                      matriz{l_matriz2+1}.valores(nframe)=labels{i}.
                            segmenttoken;
69                  end
70              end
71          end
72      end
73
74      if cont>1
75          save([d.folder '\files\etiquetas.mat'],'matriz');
76      end
77
78      guidata(hObject,d);
```

Para realizar esta propagación se halla el centro geodésico con la siguiente función.

```
1   function [centro_x,centro_y,tam2]=hallar_centro(segmentations,tok,tam1)
2       [lx,ly]=size(segmentations);
3       mat=zeros(lx,ly);
4       mat(segmentations==tok)=1;%se crea la máscara a 1's de la región
5       tam2=sum(sum(mat));
6
7       if tam2>tam1+10000%si ha crecido es que el objeto ha salido
8           centro_x=0;
9           centro_y=0;
10      else
```

```
11          D=bwdist(~mat); % imagen de distancias
12          [c_x c_y]=find(D==max(max(D)));
13          centro_x=c_x(1); % punto x del centro geodésico
14          centro_y=c_y(1); % punto y del centro geodésico
15      end
```

Por otro lado, la función que realiza la propagación de la segmentación con el detector de objetos entrantes es la siguiente.

```
1   function improve_segm_Callback(hObject, eventdata, handles)
2
3       stop_flag=0; % por si se pulsa STOP
4       handles.stop_flag=0;
5       set(handles.stop_flag,'UserData',stop_flag);
6       d = guidata(gcbo);
7       d.stop_flag=0;
8       l=importdata('list.txt');
9       pos=d.i;
10      set(d.Estado,'String','Estado: Extrapolando a...');
11      n_frames=length(l)-d.i;
12      for cont=1:n_frames
13          if exist([d.folder '\files\' l{pos} '.seg.gif'])==0
14              display('No hay archivo guardado');
15              return
16          end
17          ima1=importdata([d.folder '\files\' l{pos} '.seg.gif']);
18          ima1=ima1.cdata;
19          ima1=double(ima1);
20          ima2=importdata([d.folder '\' l{pos+1} '.segpre2.mat']);
21          ima2=getLabelImage(ima2,1); % imagen sin regiones separadas
22          ima2_r=reasignLabels(ima1,ima2); % imagen con la segmentación
                optimizada
23
24          if pos==1, % si es el frame 1 es exclusivo porque no tiene
                referencia anterior
25              %---------Frame 1---------%
26              op1=im2double(imread([d.folder '/' l{1}]));
27              op2=im2double(imread([d.folder '/' l{2}]));
28              [gmod,d.mod]=opticalflow(op1,op2,d,1);
29              [lx,ly]=size(gmod);
30              aux=gmod;
31              aux_bis=zeros(lx,ly);
32              aux_bis(1:5,:)=aux(1:5,:);aux_bis(lx-4:lx,:)=aux(lx-4:lx,:);
                    aux_bis(:,1:5)=aux(:,1:5);aux_bis(:,ly-5:ly)=aux(:,ly-5:
                    ly);
```

```
33                  suma1=sum(sum(aux_bis));
34                  %----------Frame 2----------%
35                  op1=im2double(imread([d.folder '/' l{2}]));
36                  op2=im2double(imread([d.folder '/' l{3}]));
37                  [gmod,d.mod]=opticalflow(op1,op2,d,2);
38                  [lx,ly]=size(gmod);
39                  aux=gmod;
40                  aux_bis=zeros(lx,ly);
41                  aux_bis(1:5,:)=aux(1:5,:);aux_bis(lx-4:lx,:)=aux(lx-4:lx,:);
                        aux_bis(:,1:5)=aux(:,1:5);aux_bis(:,ly-5:ly)=aux(:,ly-5:
                        ly);
42                  suma2=sum(sum(aux_bis));
43                  %----------Frame 3----------%
44                  op1=im2double(imread([d.folder '/' l{3}]));
45                  op2=im2double(imread([d.folder '/' l{4}]));
46                  [gmod,d.mod]=opticalflow(op1,op2,d,3);
47                  [lx,ly]=size(gmod);
48                  aux=gmod;
49                  aux_bis=zeros(lx,ly);
50                  aux_bis(1:5,:)=aux(1:5,:);aux_bis(lx-4:lx,:)=aux(lx-4:lx,:);
                        aux_bis(:,1:5)=aux(:,1:5);aux_bis(:,ly-5:ly)=aux(:,ly-5:
                        ly);
51                  suma3=sum(sum(aux_bis));
52
53                  suma=min(suma1,suma2);
54                  suma=min(suma,suma3);
55                  save('suma.mat','suma');
56              elseif exist('suma.mat','file')==0% si no existe referencia por
                    iniciar la propagación en n
57                  op1=im2double(imread([d.folder '/' l{pos-1}]));
58                  op2=im2double(imread([d.folder '/' l{pos}]));
59                  [gmod,d.mod]=opticalflow(op1,op2,d,pos-1);
60                  [lx,ly]=size(gmod);
61                  aux=gmod;
62                  aux_bis=zeros(lx,ly);
63                  aux_bis(1:5,:)=aux(1:5,:);aux_bis(lx-4:lx,:)=aux(lx-4:lx,:);
                        aux_bis(:,1:5)=aux(:,1:5);aux_bis(:,ly-5:ly)=aux(:,ly-5:
                        ly);
64                  suma=sum(sum(aux_bis));
65                  save('suma.mat','suma');
66              end
67              suma=load('suma.mat');
68              suma=suma.suma;
69              op1=im2double(imread([d.folder '/' l{pos}]));
70              op2=im2double(imread([d.folder '/' l{pos+1}]));
71              [gmod,d.mod]=opticalflow(op1,op2,d,pos);
```

```
72          [lx,ly]=size(gmod);
73          aux=gmod;
74          aux_bis=zeros(lx,ly);
75          aux_bis(1:5,:)=aux(1:5,:);aux_bis(lx-4:lx,:)=aux(lx-4:lx,:);
                aux_bis(:,1:5)=aux(:,1:5);aux_bis(:,ly-5:ly)=aux(:,ly-5:ly);
76          ima2_r=getLabelImage(ima2_r,1);
77          imwrite(ima2_r,[d.folder '\files\' l{pos+1} '.seg.gif']);
78          d.i=pos;
79          if exist([d.folder '\files\' l{pos} '.labels.mat']) % se propagan
                las etiquetas
80              extrapolar(hObject, eventdata, d);
81          end
82          if sum(sum(aux_bis))>round(suma*3.5) % si el valor de movimiento
                ha aumentado 3.5 veces entra objeto
83              d.i=pos+1;
84              suma=sum(sum(aux_bis));
85              save('suma.mat','suma');
86
87              d.i=pos+1;
88              set(d.Estado,'String','Estado: Detectado objeto entrante');
89              display('Detectado objeto entrante')
90              d.labels = struct([]);
91              d.polygons = struct([]);
92              d.segmentation = struct([]);
93              guidata(hObject,d);
94              initialize_image(gcbo,d.i,handles.folder);
95              return
96          else
97              set(d.Estado,'String',['Estado: Extrapolado a frame n° ',
                    num2str( pos+1 )]);
98              display(['Frame ' int2str(pos+1)])
99              d.labels = struct([]);
100             d.polygons = struct([]);
101             d.segmentation = struct([]);
102             guidata(hObject,d);
103             initialize_image(gcbo,d.i+1,handles.folder);
104             pause(0.1)
105             d = guidata(gcbo);
106             stop_flag=get(handles.stop_flag, 'UserData');
107             if stop_flag==1 % si se ha pulsado STOP
108                 stop_flag=0;
109                 set(handles.stop_flag,'UserData',stop_flag);
110                 set(d.Estado,'String','Estado: Aplicación detenida');
111                 display('Aplicación detenida');
112                 suma=sum(sum(aux_bis));
113                 save('suma.mat','suma');
```

```
                    return
                end
            end
            suma=sum(sum(aux_bis));
            save('suma.mat','suma');
            if cont==n_frames, delete('suma.mat');end
            pos=pos+1;
        end
```

Esta función como se ha visto llama a varias funciones.

La función que genera la imagen sin regiones separadas en el espacio es la siguiente.

```
function Label = getLabelImage(ima, dim)

    [~,~,c]= unique(reshape(ima,numel(ima(:,:,1)),dim),'rows'); % se
        extraen los valores de las regiones
    Label = (reshape(c,size(ima(:,:,1))));
    next = max(Label(:))+1;
    v = unique(Label);
    for j=1:numel(v),
        mk = Label == v(j);
        [BW] = bwlabel(mk);
        if max(BW(:)) ~= 1; % si hay más de una región con ese valor se
            le cambia a una
            for et = 2:max(BW(:)),
            Label(BW ==et) = next;
            next = next+1;
            end
        end
    end
```

La función que genera la imagen de segmentación optimizada es la siguiente.

```
function  ima2r = reasignLabels(ima1,ima2)

    sz = accumarray(ima2(:),ones(numel(ima2),1),[],@sum);
    valores = unique(ima1);
    ima2r = zeros(size(ima2));
    for j=1:length(valores)
        token=valores(j);
        mask = ima1 == token; % máscara de una región de la imagen a
            propagar
        szom = accumarray(ima2(:),mask(:),[],@sum);
        rt = szom./sz;
```

```
11          in = rt >= 0.5; % si está en más del 50 por ciento de esa región
                se une a las demás
12          ima2r(in(ima2)) = token;
13      end
```

Para hallar la imagen del movimiento entre dos frames se usa la siguiente función.

```
1   function [gmod,mod]=opticalflow(im1,im2,d,pos)
2       %% call optical flow
3       addpath('mex');
4       % set optical flow parameters (see Coarse2FineTwoFrames.m for the
            definition of the parameters)
5       alpha = 0.012;
6       ratio = 0.75;
7       minWidth = 20;
8       nOuterFPIterations = 7;
9       nInnerFPIterations = 1;
10      nSORIterations = 30;
11      para = [alpha,ratio,minWidth,nOuterFPIterations,nInnerFPIterations,
            nSORIterations];
12
13      tam=50;
14      vx = zeros(size(im1,1),size(im1,2)); vy = vx;
15      % en cada borde de la imagen
16      % Q1:
17      [aux_x,aux_y,~] = Coarse2FineTwoFrames(im1(:,1:tam,:),im2(:,1:tam,:)
            ,para);
18      vx(:,1:tam) = aux_x; vy(:,1:tam) = aux_y;
19      % Q2:
20      [aux_x,aux_y,~] = Coarse2FineTwoFrames(im1(1:tam,:,:),im2(1:tam,:,:)
            ,para);
21      vx(1:tam,:) = aux_x; vy(1:tam,:) = aux_y;
22      % Q3:
23      [aux_x,aux_y,~] = Coarse2FineTwoFrames(im1(:,end-tam+1:end,:),im2(:,
            end-tam+1:end,:),para);
24      vx(:,end-tam+1:end) = aux_x; vy(:,end-tam+1:end) = aux_y;
25      % Q4:
26      [aux_x,aux_y,~] = Coarse2FineTwoFrames(im1(end-tam+1:end,:,:),im2(
            end-tam+1:end,:,:),para);
27      vx(end-tam+1:end,:,:) = aux_x; vy(end-tam+1:end,:,:) = aux_y;
28
29      mod = sqrt(vx.^2 + vy.^2);
30
31      if pos==1
32          gmod = mod;
33          d.mod = mod;
```

```
34      else
35          l=importdata('list.txt');
36          if isfield(d,'mod')==0% si no existe la referencia del anterior
                hay que hallarla
37              op1=im2double(imread([d.folder '/' l{pos-1}]));
38              op2=im2double(imread([d.folder '/' l{pos}]));
39              tam=50;
40              vx = zeros(size(op1,1),size(op1,2)); vy = vx;
41              %Q1:
42              [aux_x,aux_y,~] = Coarse2FineTwoFrames(op1(:,1:tam,:),op2
                    (:,1:tam,:),para);
43              vx(:,1:tam) = aux_x; vy(:,1:tam) = aux_y;
44              %Q2:
45              [aux_x,aux_y,~] = Coarse2FineTwoFrames(op1(1:tam,:,:),op2(1:
                    tam,:,:),para);
46              vx(1:tam,:) = aux_x; vy(1:tam,:) = aux_y;
47              %Q3:
48              [aux_x,aux_y,~] = Coarse2FineTwoFrames(op1(:,end-tam+1:end
                    ,:),op2(:,end-tam+1:end,:),para);
49              vx(:,end-tam+1:end) = aux_x; vy(:,end-tam+1:end) = aux_y;
50              %Q4:
51              [aux_x,aux_y,~] = Coarse2FineTwoFrames(op1(end-tam+1:end
                    ,:,:),op2(end-tam+1:end,:,:),para);
52              vx(end-tam+1:end,:,:) = aux_x; vy(end-tam+1:end,:,:) = aux_y
                    ;
53              pmod = sqrt(vx.^2 + vy.^2);
54          else
55              pmod = d.mod;
56          end
57          gmod = sqrt((mod - pmod).^2);
58          d.mod = mod;
59      end
```

Para realizar la recuperación de la etiqueta que se ha perdido respecto a un frame anterior se usa la siguiente función.

```
1   function rescue_label_Callback(hObject, eventdata, handles)
2
3       d = guidata(gcbo);
4       l=importdata('list.txt');
5
6       if exist([d.folder '\files\' l{d.i-1} '.labels.mat'])
7           labels_prev=load([d.folder '\files\' l{d.i-1} '.labels.mat']);
8           labels_prev=labels_prev.labels;%etiquetas del frame anterior
9       else
```

```
10          display('No hay etiquetas anteriores');
11          return;
12      end
13
14      if exist([d.folder '\files\' l{d.i} '.labels.mat'])
15          labels=load([d.folder '\files\' l{d.i} '.labels.mat']);
16          labels=labels.labels;%etiquetas del frame actual
17          vacio=0;%flag indicando que no está vacío
18      else
19          labels{1,1}.label=labels_prev{1,1}.label;
20          labels{1,1}.label='vacio';
21          vacio=1;%flag indicando que está vacío
22      end
23
24      seg=imread([d.folder '\files\' l{d.i} '.seg.gif']);
25
26      long1=length(labels);
27      long2=length(labels_prev);
28      cont=0;
29
30      for y=1:long2
31          for x=1:long1
32                  %se van comparando todas las etiquetas de un frame y
                        otro
33              if strcmp(labels_prev{1,y}.label,labels{1,x}.label)==1
34                  %se comprueba que no ha variado en exceso su tamaño y
                        sigue siendo el mismo objeto
35                  if (labels_prev{1,y}.tam>labels{1,x}.tam-1000 &&
                        labels_prev{1,y}.tam<labels{1,x}.tam+1000)
36                      cont=1;
37                  end
38              end
39          end
40          if (cont==0)%no ha encontrado coincidencia con esa etiqueta
41              if vacio==1
42                  long1=0;
43              end
44              %se genera esa etiqueta anterior en el actual
45              labels{1,long1+1}=labels_prev{1,y};
46              labels{1,long1+1}.segmenttoken=seg(labels_prev{1,y}.centro_x
                    ,labels_prev{1,y}.centro_y);
47              [labels{1,long1+1}.centro_x,labels{1,long1+1}.centro_y,~]=
                    hallar_centro(seg,labels{1,long1+1}.segmenttoken,Inf);
48          end
49          cont=0;
50      end
```

```
51
52      save([d.folder '\files\' l{d.i} '.labels.mat'],'labels');
53      d.labels = struct([]);
54      d.polygons = struct([]);
55      d.segmentation = struct([]);
56      guidata(hObject,d);
57      initialize_image(gcbo,d.i,d.folder);
```

El flag que se comprueba de si se ha pulsado STOP se actualiza con la siguiente función.

```
1   function Stop_Callback(hObject, eventdata, handles)
2
3           d = guidata(gcbo);
4           stop_flag=1;%se activa el flag
5           set(handles.stop_flag,'UserData',stop_flag);
6           guidata(hObject,d);
```

Para visualizar el vídeo siguiendo una etiqueta se usa la siguiente función.

```
1   function view_all_Callback(hObject, eventdata, handles)
2
3       d = guidata(gcbo);
4       if exist([d.folder '\files\etiquetas.mat'],'file')==0%no hay
            etiquetas
5           display('No existe ninguna etiqueta');
6           return
7       end
8       l=importdata('list.txt');
9       matriz=load([d.folder '\files\etiquetas.mat']);
10      matriz=matriz.matriz;
11
12      for i=1:length(matriz)
13          categorias{1,i}=matriz{i}.etiqueta;
14      end
15
16      [sel,ok]=listdlg('ListString',categorias,'SelectionMode','single',
            ...
17          'ListSize',[250 500],'Name','select label');
18      if ok ~= 1, return, end %user clicked 'cancel'
19          %se recoge la etiqueta elegida por el usuario
20      answer=getHierarchicalLabel(categorias,sel);
21
22      for i=1:length(matriz)
23          if strcmp(matriz{i}.etiqueta,answer)==1
```

```
24              valores=matriz{i}.valores;%se recogen los valores de la
                    etiqueta en el vídeo
25          end
26      end
27
28      i_aux=d.i;
29      for z=1:d.length
30          stop_flag=0;%por si detiene el vídeo
31          handles.stop_flag=0;
32          set(handles.stop_flag,'UserData',stop_flag);
33          d = guidata(gcbo);
34          d.stop_flag=0;
35          d.i = z;
36          [lx,ly]=size(d.segmentation);
37          segmentation = load_segmentation(l{d.i}, d.mode, d.i, d.folder);
                %se carga la segmentación
38          d.selstatus = zeros(lx,ly,'uint8');
39          d.selstatus(segmentation==valores(d.i))=1;%se selecciona la
                región
40          d.labels = struct([]);
41          d.polygons = struct([]);
42          d.segmentation = struct([]);
43          guidata(hObject,d);
44          initialize_image(gcbo,d.i,handles.folder);
45          pause(0.1)
46
47          stop_flag=get(handles.stop_flag, 'UserData');%se comprueba si
                pulsa STOP
48              if stop_flag==1
49                  stop_flag=0;
50                  set(handles.stop_flag,'UserData',stop_flag);
51                  display('Aplicación detenida');
52                  return
53              end
54
55          if z==d.length%si es el último frame se muestra por donde lo
                dejó el usuario
56                  i=i_aux;
57                  d.labels = struct([]);
58                  d.polygons = struct([]);
59                  d.segmentation = struct([]);
60                  guidata(hObject,d);
61                  initialize_image(gcbo,i,handles.folder);
62          end
63      end
```

Además de estas funciones se han generado dos para facilitar el trabajo y borrar los ficheros automáticamente.

En esta función se elimina la segmentación inicial.

```
function delete_segm(folder)
        %ppm si es Felzenszwalb y mat si es superpíxels
        %system(['dir .\' folder '\*.ppm /b  > borrar.txt']);

    system(['dir .\' folder '\*.mat /b  > borrar.txt']);
    l=importdata('borrar.txt');

    k=length(l);

    for z=1:k
        delete([folder '/' l{z}])
    end
    delete('borrar.txt')
end
```

Por otro lado, para borrar los ficheros que se generan con las etiquetas y las segmentaciones se usa la siguiente función.

```
function delete_files(folder)
        %fichero de etiquetas
    system(['dir .\' folder '\' 'files' '\*.mat /b  > borrar.txt']);
    l=importdata('borrar.txt');
    k=length(l);
    for z=1:k
        delete([folder '/' 'files' '/' l{z}])
    end
    %fichero de segmentación
    system(['dir .\' folder '\' 'files' '\*.gif /b  > borrar.txt']);
    l=importdata('borrar.txt');
    k=length(l);
    for z=1:k
        delete([folder '/' 'files' '/' l{z}])
    end
    delete('borrar.txt')
end
```